

Tutorial 5: Texture Mapping

by Victor Saar

Content

Now where we have done all the basic stuff, like the initialization of Direct3D and creation of simple objects, we can make something new. In this tutorial we will start using texture mapping. Textures are a simple 3D effect, but have an enormous result and are very easy to use. The first thing we have to do is to change the custom vertex format. Then we only need to load the texture with help of the Direct3DX utility library. Two important things are the texture coordinates, which we need to establish how the texture is displayed on the triangle, and the filters, which smooth the texture. The filters are already set in the application class. You can choose the filter type in the settings dialog.

d3ddefs.h

Here we change the vertex format. First we define the constant, which shows how the vertex format is constructed. We define a vertex with a x, y and z coordinate and one set of texture coordinates.

```
#define D3DFVF_CUSTOMVERTEX (D3DFVF_XYZ | D3DFVF_TEX1)
```

Here is the vertex structure with the coordinates. The `fU` and `fV` coordinates are for the texture.

```
struct D3DVERTEX
{
    float fX,
          fY,
          fZ;
    float fU, //texture coordinates (NEW)
          fV;
};
```

main.cpp

We need one more object for the texture here.

```
LPDIRECT3DTEXTURE9 pPyramideTexture = NULL; //texture object (NEW)
```

Now we have to change the vertex information for the polygons. The last two values of each vertex in the structures are the texture coordinates.

```
D3DVERTEX aTriangle[] = {{ 0.0f, 1.0f, 0.0f, 0.5f, 0.0f},
                          {-1.0f, -1.0f, -1.0f, 0.0f, 1.0f},
                          { 1.0f, -1.0f, -1.0f, 1.0f, 1.0f},
                          { 0.0f, 1.0f, 0.0f, 0.5f, 0.0f},
                          {-1.0f, -1.0f, 1.0f, 0.0f, 1.0f},
                          {-1.0f, -1.0f, -1.0f, 1.0f, 1.0f},
                          { 0.0f, 1.0f, 0.0f, 0.5f, 0.0f},
                          { 1.0f, -1.0f, 1.0f, 0.0f, 1.0f},
                          {-1.0f, -1.0f, 1.0f, 1.0f, 1.0f},
                          { 0.0f, 1.0f, 0.0f, 0.5f, 0.0f},
                          { 1.0f, -1.0f, -1.0f, 0.0f, 1.0f},
                          { 1.0f, -1.0f, 1.0f, 1.0f, 1.0f}};

D3DVERTEX aQuad[] = {{-1.0f, -1.0f, -1.0f, 0.0f, 0.0f},
```

```
{ 1.0f,-1.0f,-1.0f,1.0f,0.0f},  
{-1.0f,-1.0f, 1.0f,0.0f,1.0f},  
{ 1.0f,-1.0f, 1.0f,1.0f,1.0f}};
```

After we have filled the vertex buffers with the vertices, we must load the texture from a file. We use the Direct3DX function `D3DXCreateTextureFromFile()`. The parameters are the device, the name of the file, that holds the texture and the texture object. Then we have to tell Direct3D that we want to use the texture with the function `SetTexture()`.

```
//load texture from file (NEW)  
D3DXCreateTextureFromFile(g_App.GetDevice(),  
                        "texture.png",  
                        &pPyramideTexture);  
  
//set texture (NEW)  
g_App.GetDevice()->SetTexture(0,pPyramideTexture);
```

The rest of the project needn't to be changed. The texture is now mapped on the polygons of the pyramide.