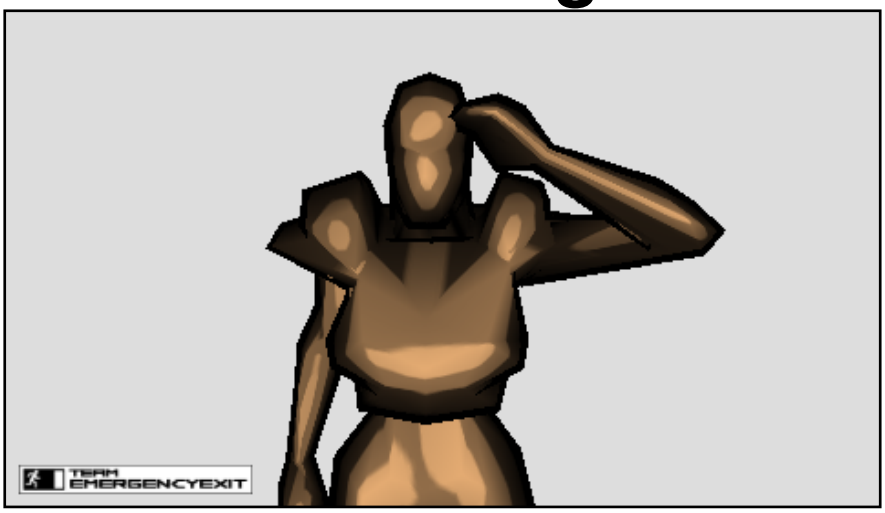
Team Emergency Exit
PSP Cel Shading Tutorial
by McZonk

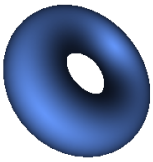
PSP Cel Shading Tutorial



Basics
The celshading look is based on two concepts. A stepped lightmap and an outline around the model. So let's begin with drawing some simple model. A torus is never a bad idea.

Stepped lightmap

This is the result when you draw a torus without any lighting or shading. Just a boring circle with a hole in it. With some diffuse lighting you can add much more atmosphere.

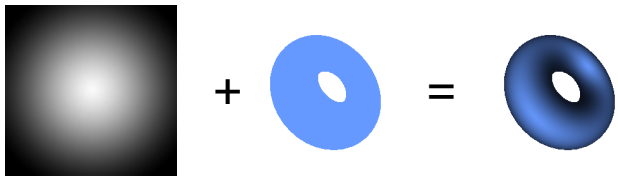


The idea is a stepped lightmap. The normal lightmap for diffuse lighting looks like this. A smooth gradient from dark to bright. Where the angle of light on the surface is zero the full brightness is combinded with the model's color. Where the angle gets bigger the brighness gets lower.

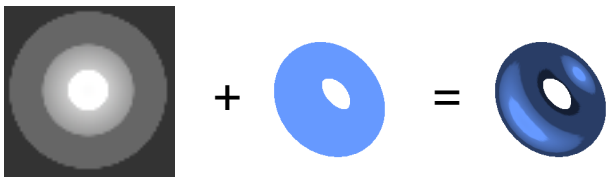
Team Emergency Exit
PSP Cel Shading Tutorial
by McZonk

The second map is a lightmap for cel shading look. Instead of being smooth it is diveded in some grads. For a good celshading look, you should use have 3 to 6 divisions in the lightmap.

A problem is that the psp doesn't support 1d lightmaps. But instead of use a 1d lightmap enviroment maps can also be used. The same effect as a diffuse lighting can be achieved with this enviroment map.



The same works with an cel shading lightmap built as a circle.



There is a nice sample that demonstres environment maps on the psp has been written by chp from www.pspdev.org. Look at this sample to see how to work with environment maps.

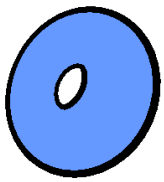
```
ScePspFVector3 envmapMatrixColumns[2] = {
    { 1.0f, 0.0f, 0.0f },
    { 0.0f, 1.0f, 0.0f }
};
sceGuLight( 0, GU_DIRECTIONAL, GU_DIFFUSE, &envmapMatrixColumns[0] );
sceGuLight( 1, GU_DIRECTIONAL, GU_DIFFUSE, &envmapMatrixColumns[1] );

sceGuTexMode(GU_PSM_8888, 0, 0, 0);
sceGuTexImage(0, 64, 64, 64, lightmaptexture);
sceGuTexFunc(GU_TFX_MODULATE, GU_TCC_RGB);
sceGuTexFilter(GU_LINEAR, GU_LINEAR);

sceGuTexMapMode(GU_ENVIRONMENT_MAP, 0, 1); The
```

This is the code snippet I used to make cel shading. The following figure is how it should look in the end.

Outlines

The second pass in cel shading is the outline around the model. This can be achieved by making the model a little bit thicker and draw the backside again. To draw only the backside you need to use culling. There are

two different ways availiable. First draw the backside wireframe with thick lines or scale up the model along the normals. On the psp only the second ways is possible because thicker lines are not supported.



To calculate the scaled up model, you need to use this for every vertex in the model.

```
unsigned int n;
for(n = 0; n < model->vertex_count; n++) {
    scaled_model->vertex[n].x = model->vertex[n].x + model->normal[n].x * OUTLINE_STRENGTH;
    scaled_model->vertex[n].y = model->vertex[n].y + model->normal[n].y * OUTLINE_STRENGTH;
    scaled_model->vertex[n].z = model->vertex[n].z + model->normal[n].z * OUTLINE_STRENGTH;
}
```

The new model doesn't need normales or texture coordinates. Draw the model by setting the color to black, flip the frontside and disable everything that is not needed like lighting and textures. To draw the model, use this code.

```
sceGuDisable(GU_LIGHTING);
sceGuDisable(GU_TEXTURE_2D);
sceGuFrontFace(GU_CCW);
sceGuColor(0x00000000);

sceGumDrawArray(GU_TRIANGLES,
    GU_NORMAL_32BITF|GU_VERTEX_32BITF|GU_INDEX_16BIT|GU_TRANSFORM_3D,
    sizeof(torus_in)/sizeof(unsigned short), torus_in, torus_v0
);

sceGuFrontFace(GU_CW);
sceGuEnable(GU_LIGHTING);
sceGuEnable(GU_TEXTURE_2D);
```

That's all. Use the effect in some of you demos.

If you want to have textures in model you will have to add a third pass. At first render the textured object, blend the celshading map and at last draw the outline.