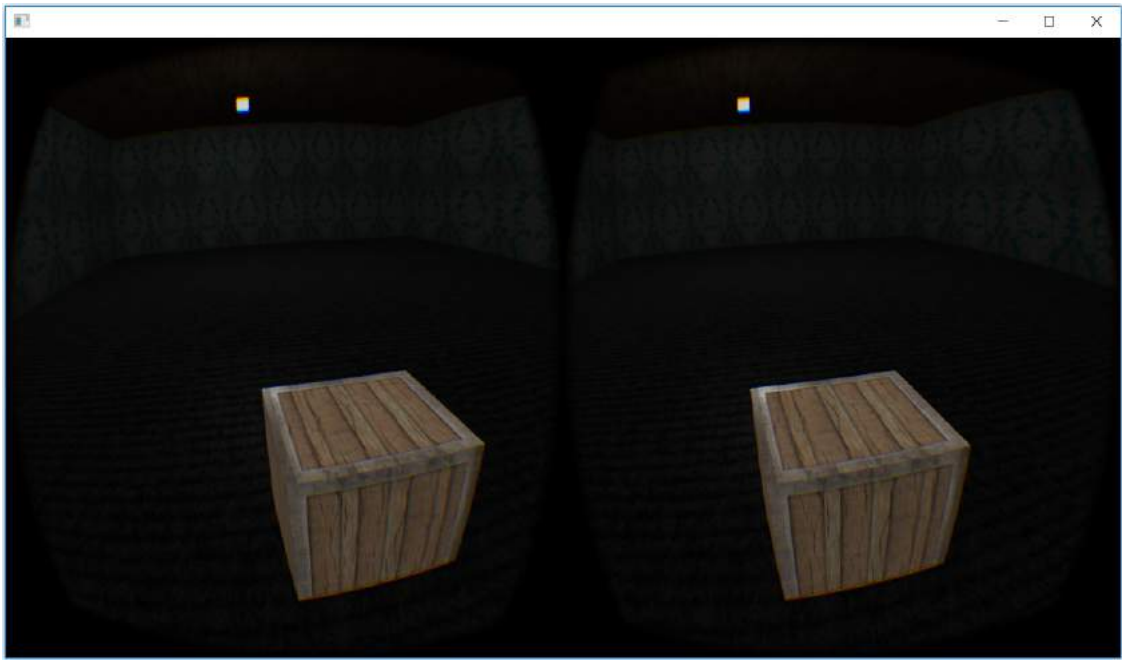
DOCUMENTACIÓN DE LA IMPLEMENTACIÓN:

Specular mapping



informatika fakultatea
facultad de informática

eman ta zabal zazu
Universidad del País Vasco
Euskal Herriko Unibertsitatea

Iker Pintado García

Índice

1. Introducción

En la vida real un objeto puede tener varios materiales diferentes a la vez, y esto conlleva a que un solo objeto no brille de la misma manera en todas sus partes. Para tratar de emular esta característica de algunos objetos, se utiliza la técnica de mapeo especular. Esto significa que, en vez de que cada objeto tenga sus propiedades de especularidad, estas vendrán dadas en un mapa para que así un solo objeto pueda tener distintos brillos.

2. Implementación

Para la implementación de esta técnica se han copiado los vertex y fragment shaders de **perfragment**. Aunque, se han aplicado algunos cambios en el fragment shader a la hora de calcular la especularidad. Lo único que se debe cambiar son todos los usos de **theMaterial.specular** e intercambiarlos por un texel de especularidad que se obtiene con el mapa especular:

```
.
.
.
void aporte_dir(in int i,in vec3 l,in vec3 n,in vec3 v,
                inout vec3 acum1, inout vec3 acum2, in vec4
           texel){
      .
      .
      .
      if (factor1 > 0.0){
            .
            .
            .
            if (factor2 > 0.0){
                  factor2= factor1 *
                              pow(factor2,theMaterial.shininess);
                  acum2 = acum2 + factor2 * texel.rgb *
                                    theLights[i].specular;
            }

      }

}
.
.
.
void aporte_pos(in int i,in vec3 l,in vec3 n,in vec3 v,
                inout vec3 acum1, inout vec3 acum2, in vec4
           texel){
      .
      .
      .
      if (factor1 > 0.0){
            .
            .
            .
            if (factor2 > 0.0){
                  factor2 = factor1 * pow(factor2,
                                          theMaterial.shininess);
                  acum2 = acum2 + atenuacion * factor2 * texel.rgb *
```

```
                              theLights[i].specular;
                }

        }

}
.
.
.
void aporte_foco(in int i,in vec3 l,in vec3 n,in vec3 v,
                 inout vec3 acum1, inout vec3 acum2, in vec4
            texel){
        .
        .
        .
        if (cosOs > 0.0){
            if (cosOs >= theLights[i].cosCutOff){
                    .
                    .
                    .
                    if (factor1 > 0.0){
                            .
                            .
                            .
                            if (factor2 > 0.0){
                                factor2 = factor1 * pow(factor2,
                                          theMaterial.shininess);
                                acum2 = acum2 + spt * factor2 *
texel.rgb
                                          * theLights[i].specular;
                            }

                    }
            }
        }

}
.
.
.
void main() {
        .
        .
        .
        vec4 texel = texture2D(specmap, f_texCoord);
        .
        .
        .
```

```
        for(int i = 0; i < active_lights_n; i++){
              if (theLights[i].position.w == 0.0){
                    L = normalize(-1.0 * theLights[i].position.xyz);
                    aporte_dir(i, L, N, V, acum_dif, acum_esp, texel);

              }else{
                    L = theLights[i].position.xyz - V_P;
                    if (theLights[i].cosCutOff == 0){
                          aporte_pos(i, L, N, V, acum_dif, acum_esp,
                                      texel);
                    }else{
                          L = normalize(L);
                          aporte_foco(i, L, N, V, acum_dif, acum_esp,
                                       texel);

                    }

              }

        }
        .
        .
        .
}
```

3. Resultado en ejecución

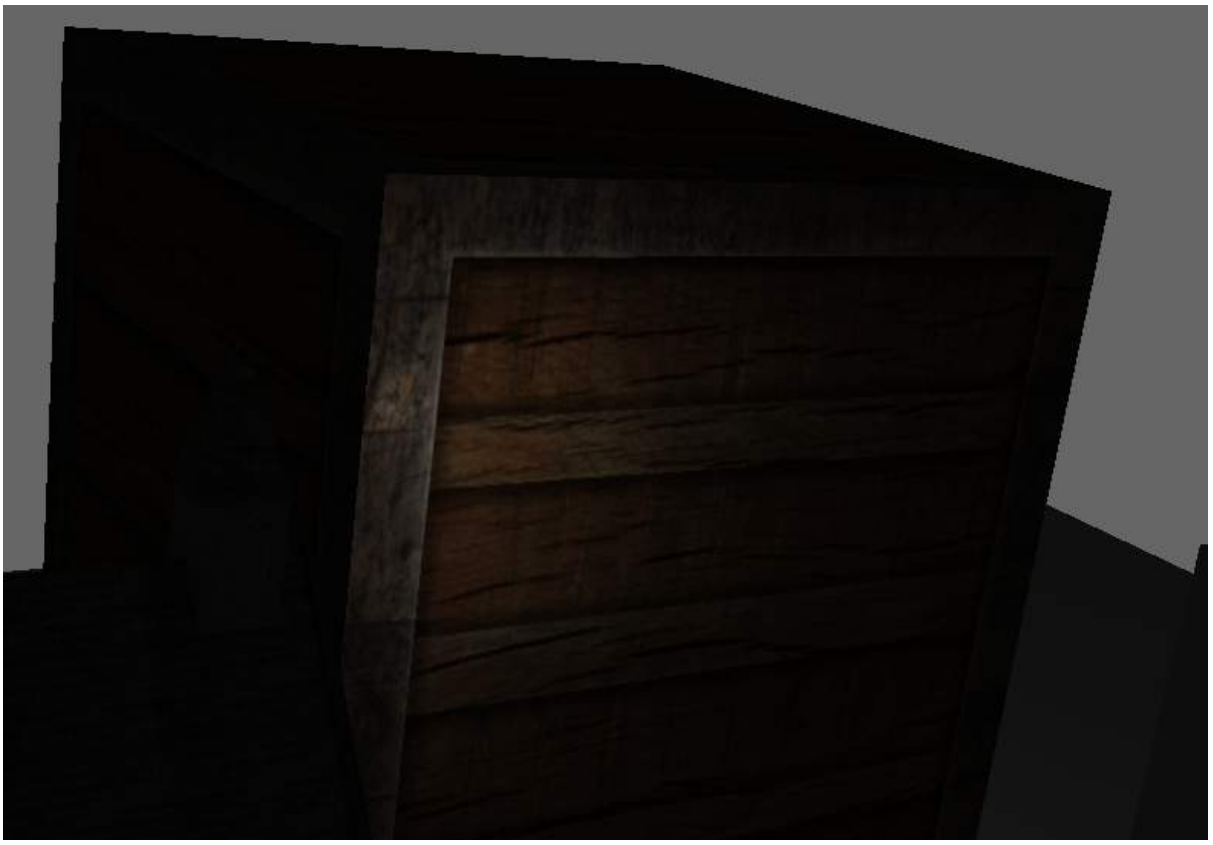


Fig. 1. Luz posiciona activada, iluminando madera y metal

Fig. 2. Caja con bordes metálicos iluminada con direccional y foco