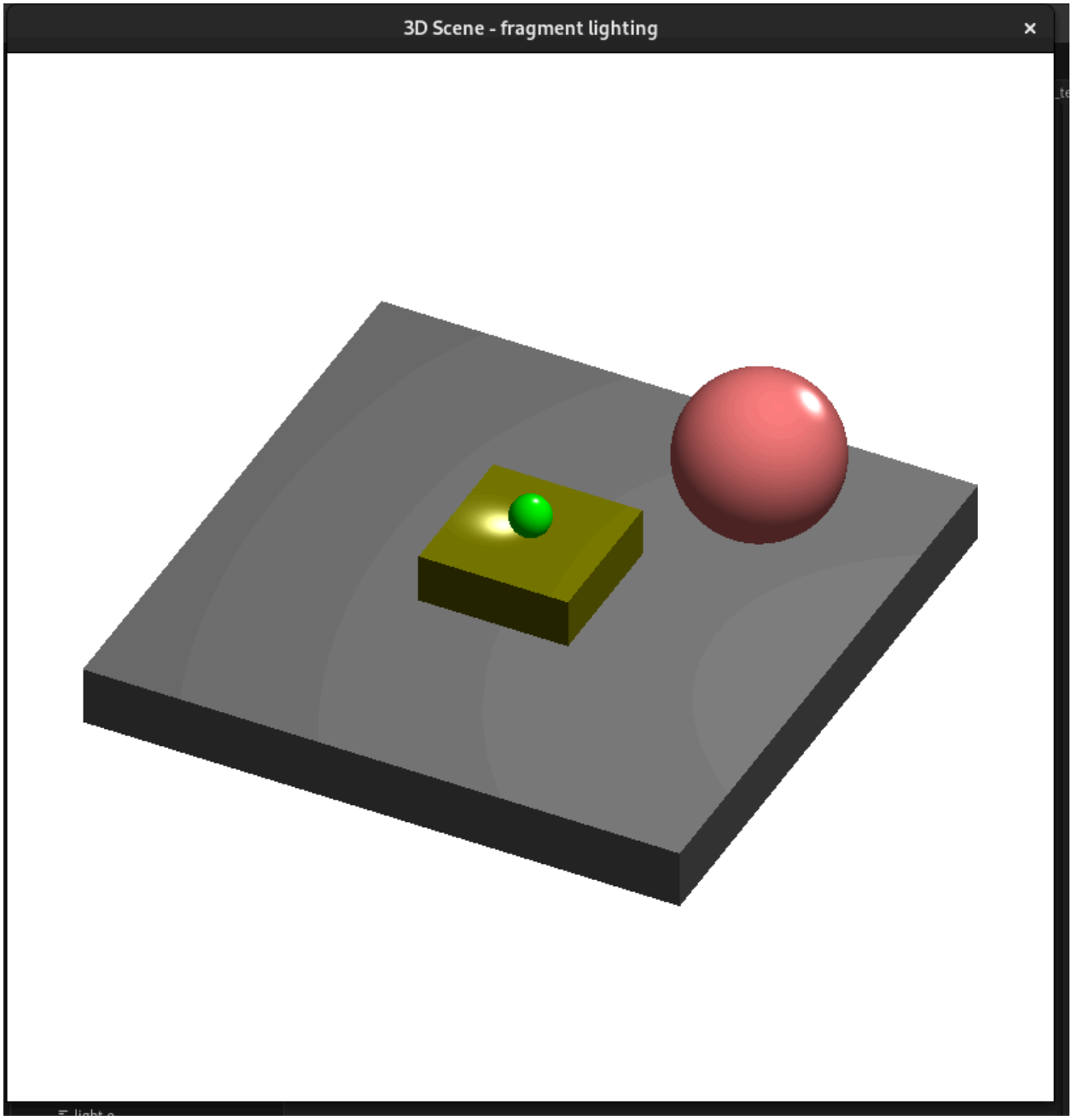
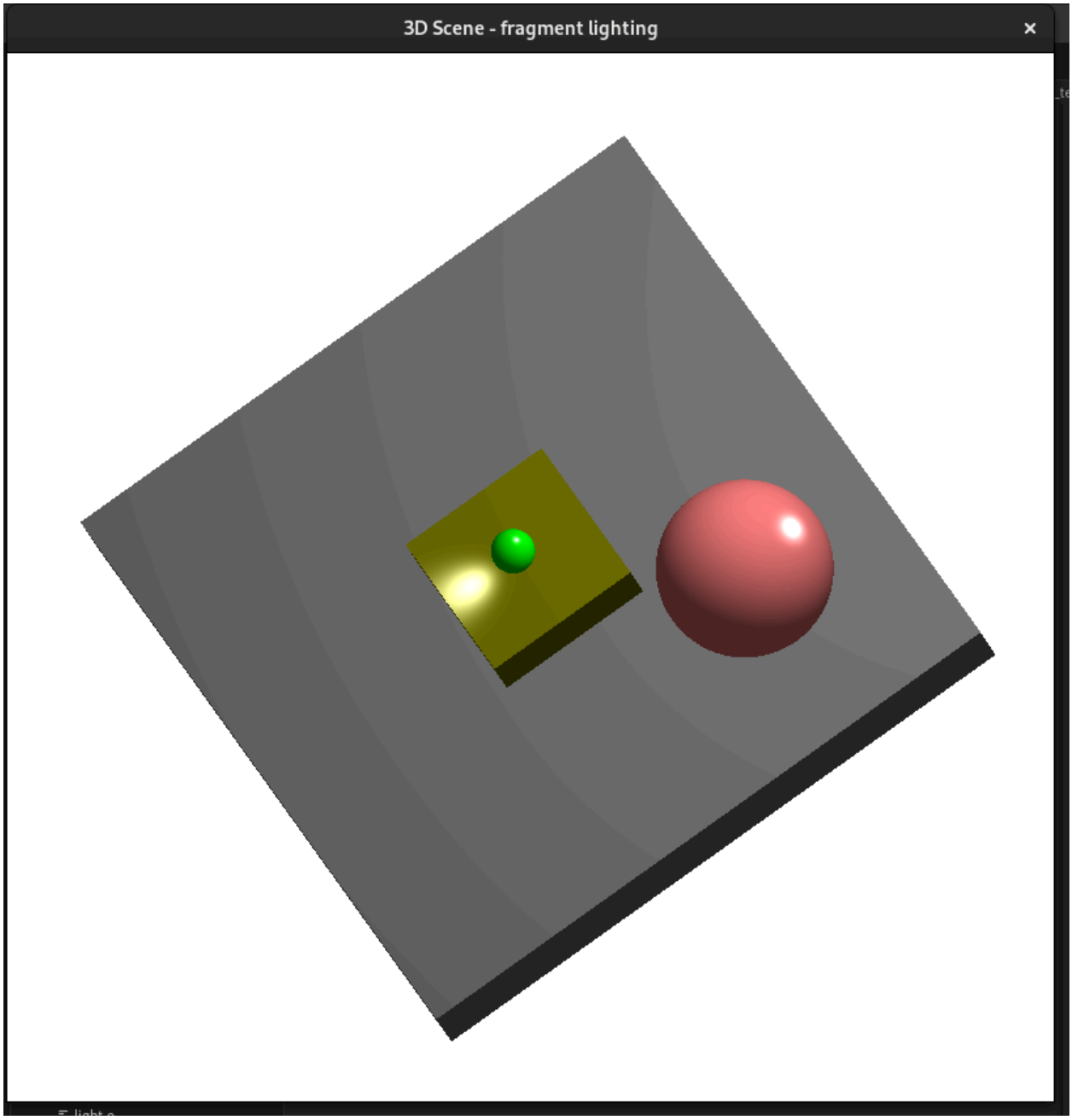
Computação Gráfica - Tarefa 2.1

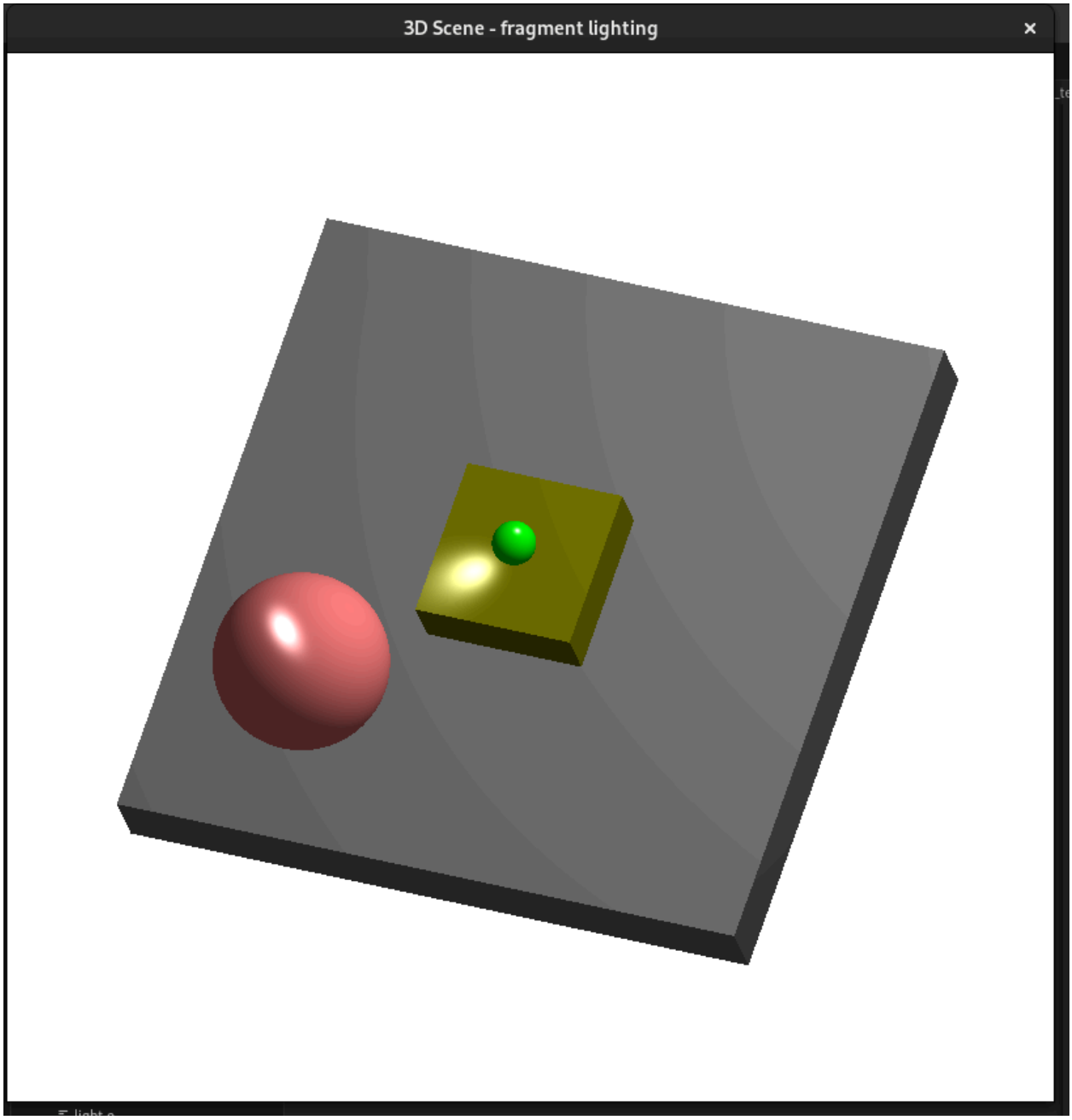
Lucas Toscano Pimentel Appolinário Cerqueira - 2110695

Screenshots obtidos utilizando a iluminação por fragmento

A diferença é mais perceptível na maneira como o bloco amarelo é iluminado. Como essa iluminação é feita por fragmento, e não por vértice, ela permite maior precisão, especialmente em objetos com poucos vértices, como este bloco.




3D Scene - fragment lighting


3D Scene - fragment lighting

Código do Vertex shader

```
#version 410

layout(location = 0) in vec4 coord;
layout(location = 1) in vec3 normal;
layout(location = 3) in vec2 texcoord;

uniform mat4 Mv;
uniform mat4 Mn;
uniform mat4 Mvp;

uniform vec4 lpos;
uniform vec4 lamb;
uniform vec4 ldif;
uniform vec4 lspe;

uniform vec4 mamb;
uniform vec4 mdif;
uniform vec4 mspe;
uniform float mshi;

out data {
  vec3 fragPos;
  vec3 fragNormal;
  vec3 fragLightDir;
} v;

void main(void)
{
  vec3 veye = vec3(Mv*coord);
  vec3 light;
  if (lpos.w == 0)
    light = normalize(vec3(lpos));
  else
    light = normalize(vec3(lpos)-veye);
  vec3 neye = normalize(vec3(Mn*vec4(normal,0.0f)));

  v.fragPos = -veye;
  v.fragNormal = neye;
  v.fragLightDir = light;

  gl_Position = Mvp * coord;
}

```

Código do Fragment shader

```
#version 410

in data {
  vec3 fragPos;
  vec3 fragNormal;
  vec3 fragLightDir;
} f;

uniform vec4 lamb;
uniform vec4 ldif;
uniform vec4 lspe;

uniform vec4 mamb;
uniform vec4 mdif;
uniform vec4 mspe;
uniform float mshi;

out vec4 fragColor;

void main(void)
{
  vec3 norm = normalize(f.fragNormal);
  vec3 lightDir = normalize(f.fragLightDir);

  // Ambient component
  vec4 ambient = mamb * lamb;

  // Diffuse component
  float ndotl = max(dot(norm, lightDir), 0.0);
  vec4 diffuse = mdif * ldif * ndotl;

  // Specular component
  vec4 specular = vec4(0.0);
  if (ndotl > 0.0) {
      vec3 viewDir = normalize(-f.fragPos);
      vec3 reflectDir = reflect(-lightDir, norm);
      float spec = pow(max(dot(viewDir, reflectDir), 0.0), mshi);
      specular = mspe * lspe * spec;
  }

  // Final color
  fragColor = ambient + diffuse + specular;

}
```