实验：
法线贴图

华中科技大学软件学院　万琳

提纲

1 实验要求

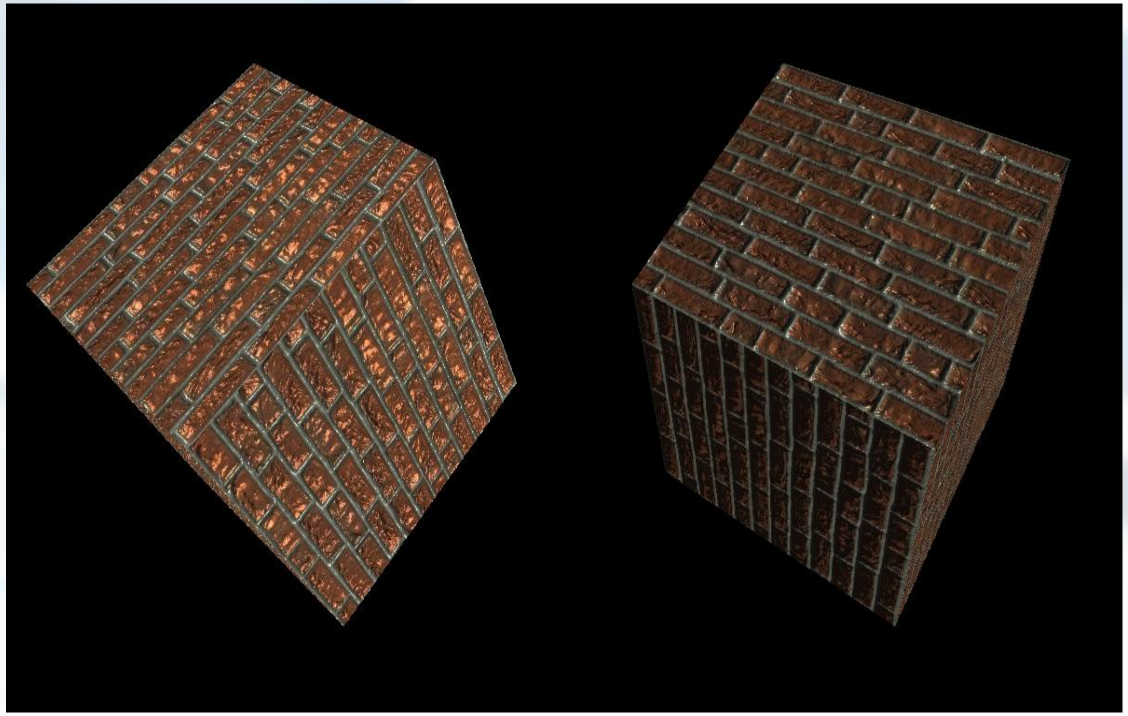
2 程序流程

3 要点解析

1

实验要求

要求：基于切线空间实现法线贴图（达到右边图的效果）



3

要点解析

➢问题分析

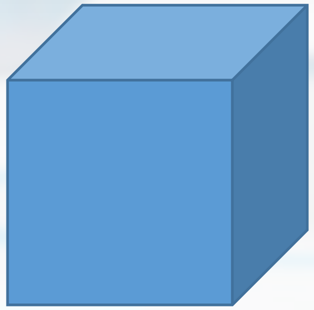
Heightmap的使用

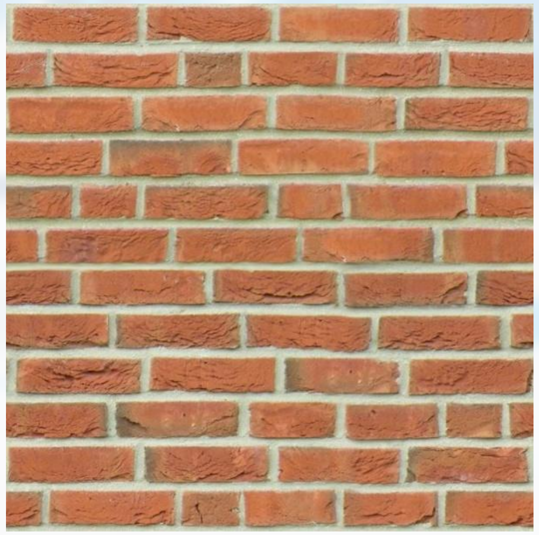
切线空间的引入

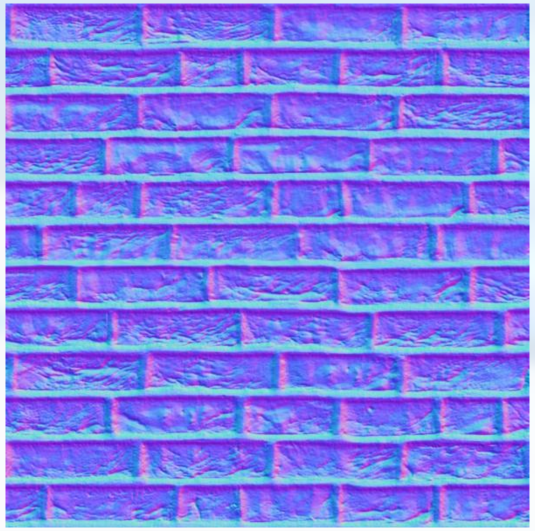
3 要点解析

➤问题一：Heightmap的使用

Heightmap的使用

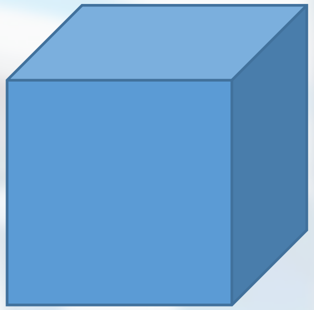


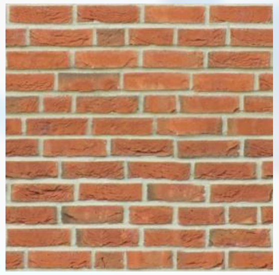

纹理图


Heightmap

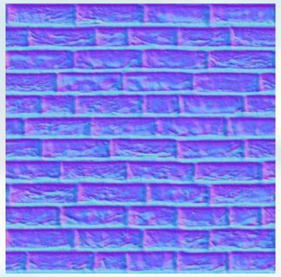
3 要点解析

➢问题一：Heightmap的使用





纹理图



Heightmap

```
GLuint cube_diffuse_texture =
LoadTextureFromFile("res/texture/cube_diffuse.jpg");//加载纹理
    GLuint cube_normal_texture =
LoadTextureFromFile("res/texture/cube_normal.jpg");//加载法线贴图

    Shader normalmap_shader("res/shader/normal.vs",
"res/shader/normal.fs");//加载着色器
```

3 **要点解析**

➢问题一：Heightmap的使用

normal.fs

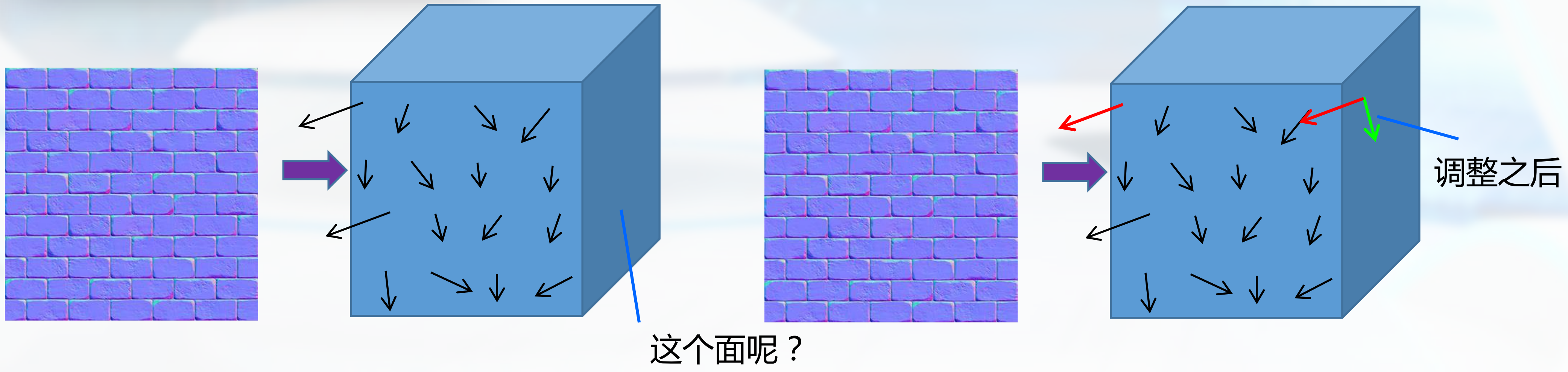
```
#version 330 coreout vec4 FragColor;
in VS_OUT{    vec3 FragPos;    vec2 TexCoords;}
fs_in;
uniform sampler2D texture_material;
uniform sampler2D texture_normal;
uniform vec3 light_direction;
uniform vec3 light_ambient;
uniform vec3 light_diffuse;
uniform vec3 light_specular;
uniform vec3 view_position;
uniform mat4 model;
void main()
{vec3 normal = texture(texture_normal, fs_in.TexCoords).rgb;
normal = normalize(normal * 2.0f - 1.0f);
vec3 view_direction = normalize(view_position - fs_in.FragPos);
```

3

要点解析

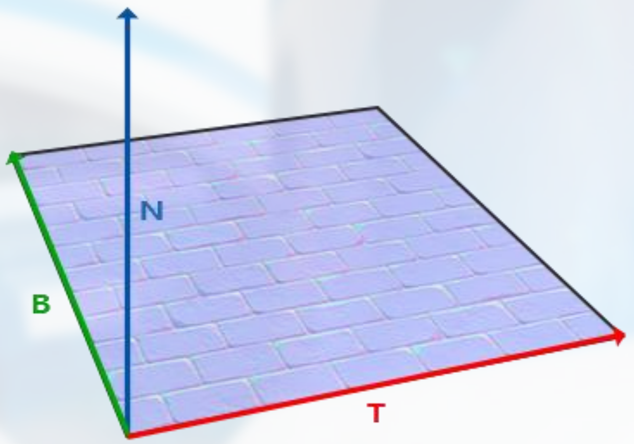
➢问题二：切线空间的引入

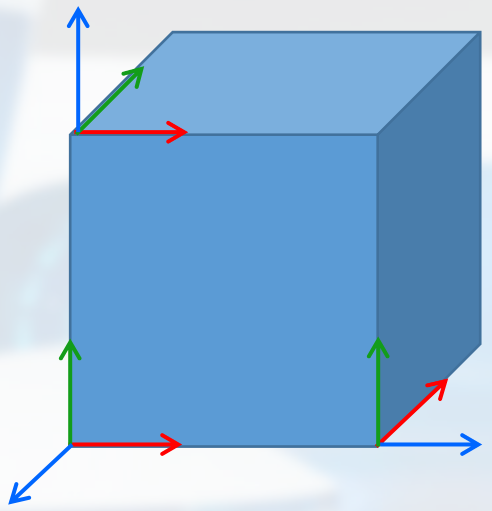
切线空间的
引入

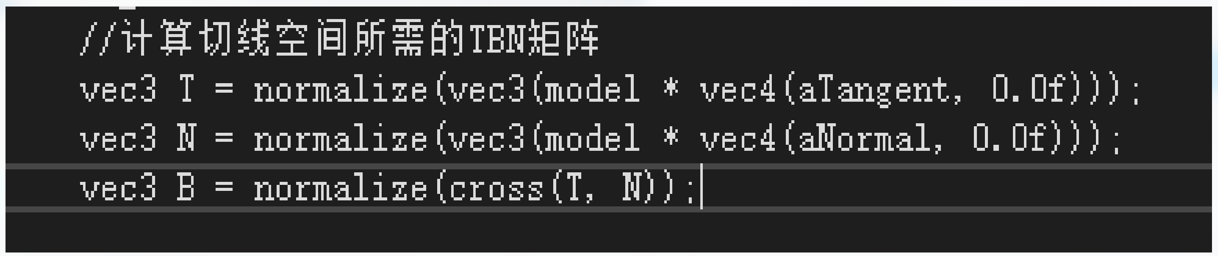


3 要点解析

➤问题二：切线空间的引入





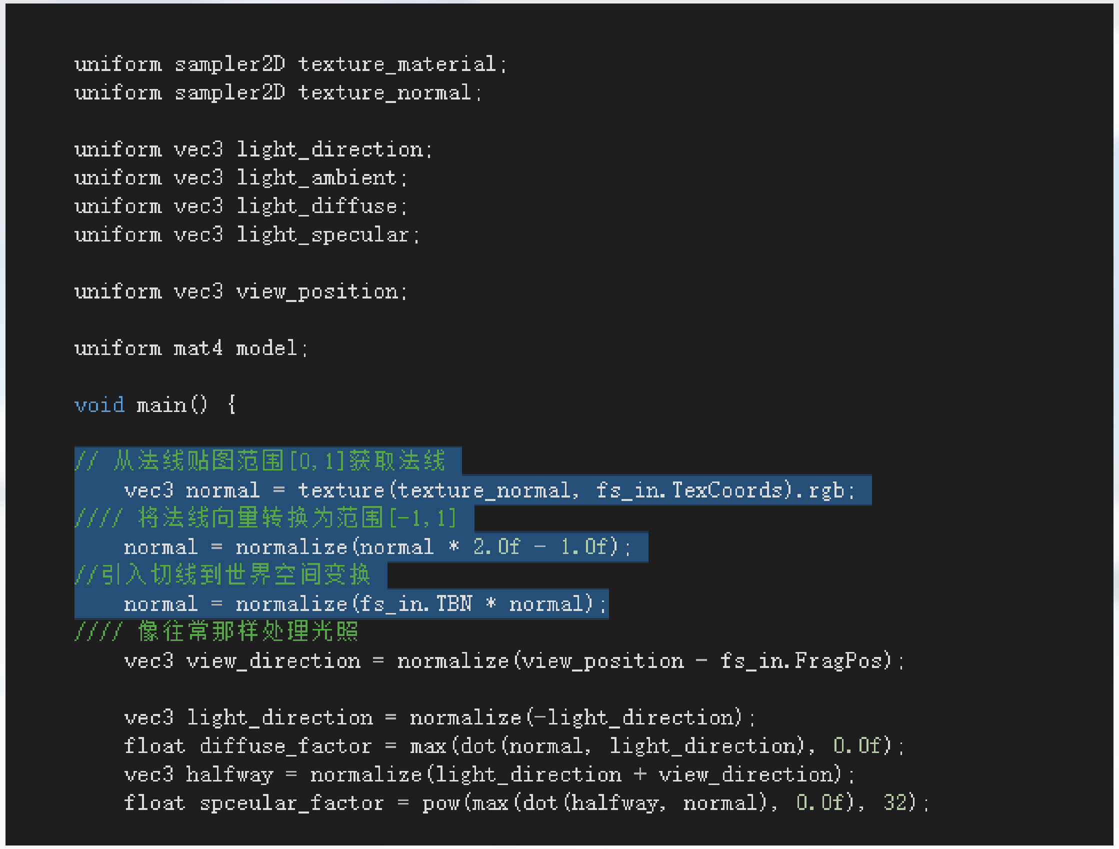


```
    //计算切线空间所需的TBN矩阵
    vec3 T = normalize(vec3(model * vec4(aTangent, 0.0f)));
    vec3 N = normalize(vec3(model * vec4(aNormal, 0.0f)));
    vec3 B = normalize(cross(T, N));
```

3

要点解析

➤问题二：切线空间的引入



```
uniform sampler2D texture_material;
uniform sampler2D texture_normal;

uniform vec3 light_direction;
uniform vec3 light_ambient;
uniform vec3 light_diffuse;
uniform vec3 light_specular;

uniform vec3 view_position;

uniform mat4 model;

void main() {

// 从法线贴图范围[0,1]获取法线
    vec3 normal = texture(texture_normal, fs_in.TexCoords).rgb;
//// 将法线向量转换为范围[-1,1]
    normal = normalize(normal * 2.0f - 1.0f);
//引入切线到世界空间变换
    normal = normalize(fs_in.TBN * normal);
//// 像往常那样处理光照
    vec3 view_direction = normalize(view_position - fs_in.FragPos);

    vec3 light_direction = normalize(-light_direction);
    float diffuse_factor = max(dot(normal, light_direction), 0.0f);
    vec3 halfway = normalize(light_direction + view_direction);
    float spceular_factor = pow(max(dot(halfway, normal), 0.0f), 32);
```

谢谢

软件学院　万琳