实验八 Shader

ID: 58 陈文博

April 18, 2020

1 实验要求

* 实现 normal map 和 displacement map
* 利用 displacement map 在 vertex shader 中进行简单降噪
* 实现点光源阴影（可选）

2 开发环境

IDE：Microsoft Visual Studio 2019 community

CMake：3.16.3

Assimp：5.0.1

GLFW：3.4.0

Others

3 算法原理

3.1 法向贴图

典型使用方式如下：

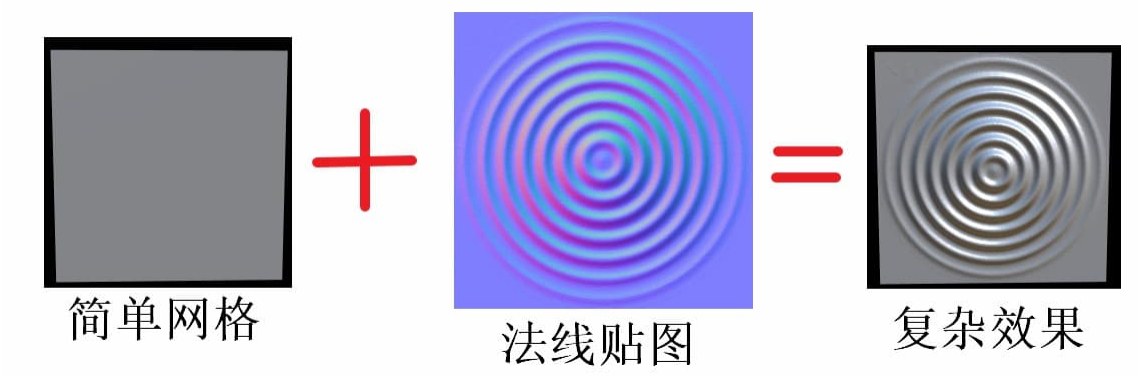

Figure 3.1: 法向贴图的使用

由于贴图中的法向是定义在切空间中的，上方向为 z 方向，对应于 RGB 的 B 通道，故法线贴图一般为蓝紫色。根据表面细微的细节对法线向量进行改变，可以获得一种表面看起来要复杂得多的幻觉：

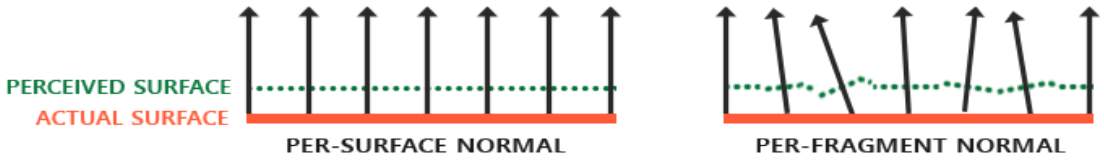

Figure 3.2: 法向贴图的原理

3.2 置换贴图

典型使用方式如下：

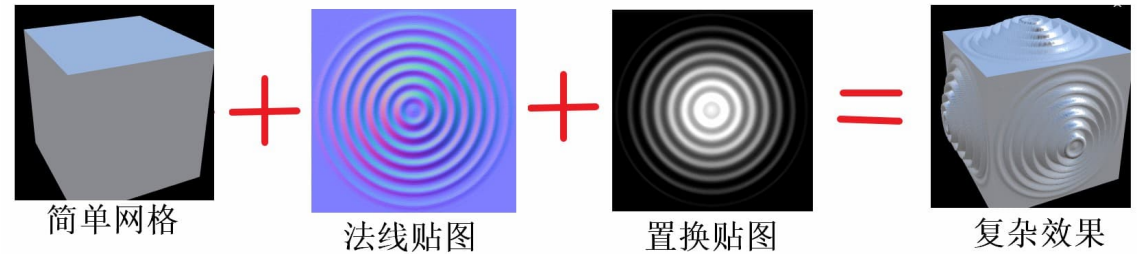

Figure 3.3: 置换贴图的使用

置换贴图中黑色 (0) 表示不动，白色 (1) 表示沿法向移动，可以定义置换贴图像素值与偏移量的关系：
displacement=lambda*(bias+scale*pixel value)

* bias=-1, scale=2: 0-0.5 为下沉，0.5-1.0 为凸起，变化量用一个系数决定
* bias=0, scale=1: 0-1.0 为凸起，变化量用一个系数决定

在实时渲染中在 vertex shader 中采样置换贴图，计算顶点偏移量来偏移顶点，同时添加相应的法向贴图，使渲染效果更加正确。
置换贴图用于降噪：

- 计算每个顶点的偏移量

$$\delta_i = p_i - \frac{1}{|N(i)|}\sum_{j\in N(i)} p_j \tag{3.1}$$

- 将偏移量投影到法向上

$$\bar{\delta}_i = \langle \delta_i, \boldsymbol{n}_i \rangle \boldsymbol{n}_i \tag{3.2}$$

- 对每一个顶点进行偏移

$$\bar{p}_i = p_i - \lambda \bar{\delta}_i = p_i - \lambda \langle \delta_i, \boldsymbol{n}_i \rangle \boldsymbol{n}_i \tag{3.3}$$

- 将 $\langle \delta_i, \boldsymbol{n}_i \rangle$ 插值存到置换贴图中

3.3 点光源阴影

3.3.1 从光源渲染出深度图

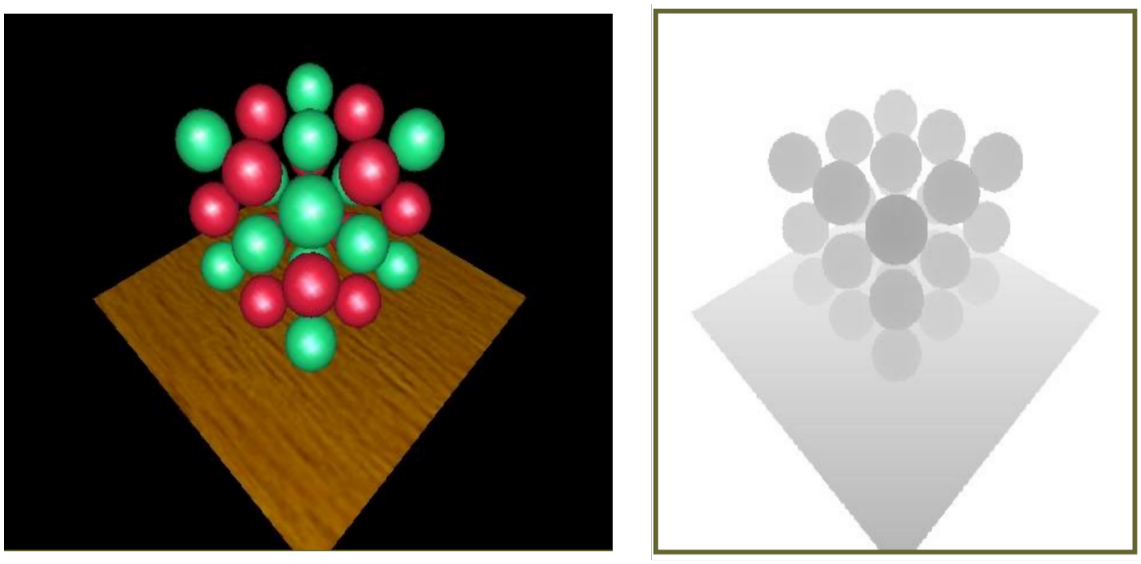

Figure 3.4: 从光源渲染

3.3.2 从摄像机渲染

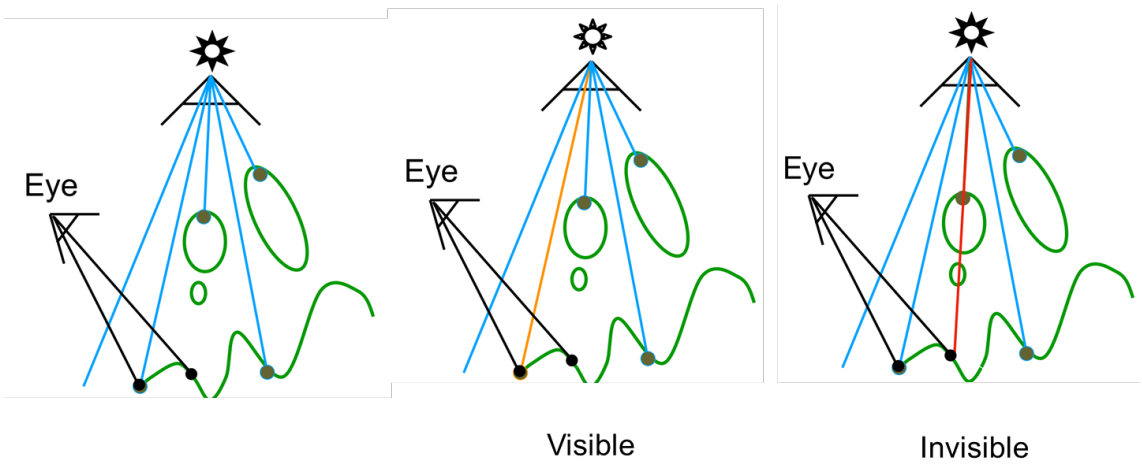

Figure 3.5: 从摄像机渲染

4 设计难点与解决

4.1 用置换贴图去噪时 displacementData 插值

使用 Homework2 中的 ANN 库进行 k 近邻插值，实际运行效果还行，但速度较慢

4.2 置换贴图去噪时出现裂痕

产生原因：由于网格拼接处相同位置的顶点采用相同的噪声，对其进行重心偏移后彼此分离造成裂痕。解决思路：遍历顶点寻找坐标相同点（衔接点）并保存，衔接点的偏移量设置为所有具有相同坐标的点偏移量的平均值。

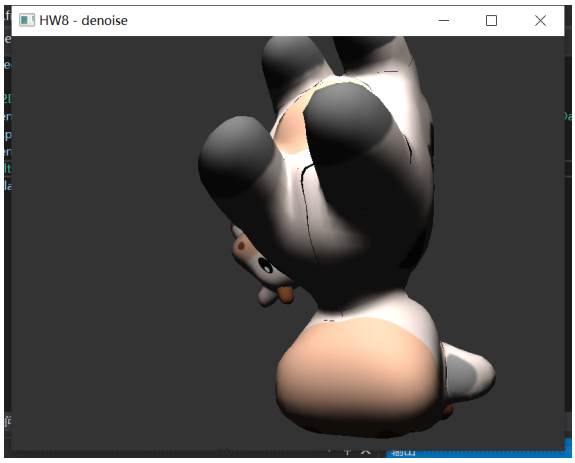

Figure 4.1: 裂痕现象

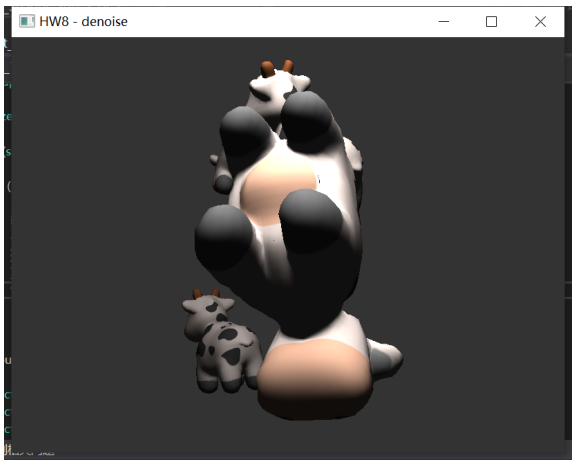

Figure 4.2: 裂痕修复

5 实验效果

5.1 法向贴图

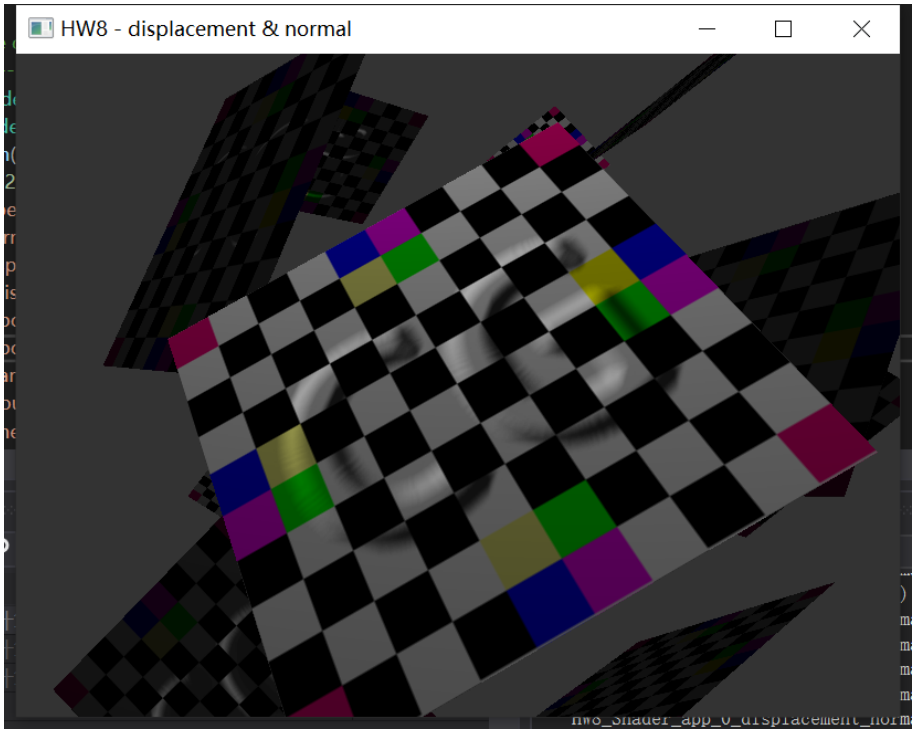

Figure 5.1: 法向贴图

5.2 置换贴图

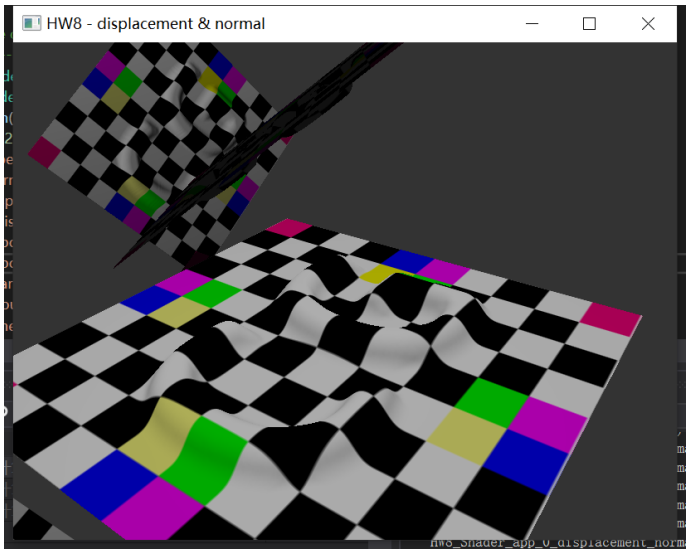

Figure 5.2: 置换贴图

其他效果：

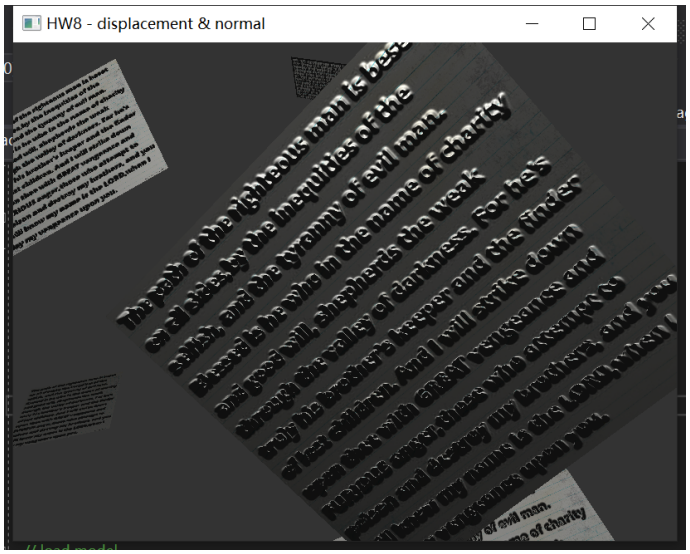

Figure 5.3: 台词贴图

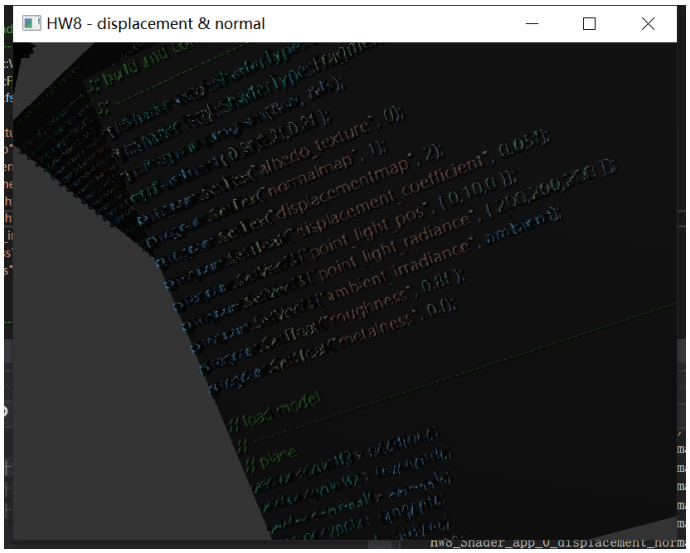

Figure 5.4: 立体代码

Figure 5.5: 校徽贴图

5.3 降噪

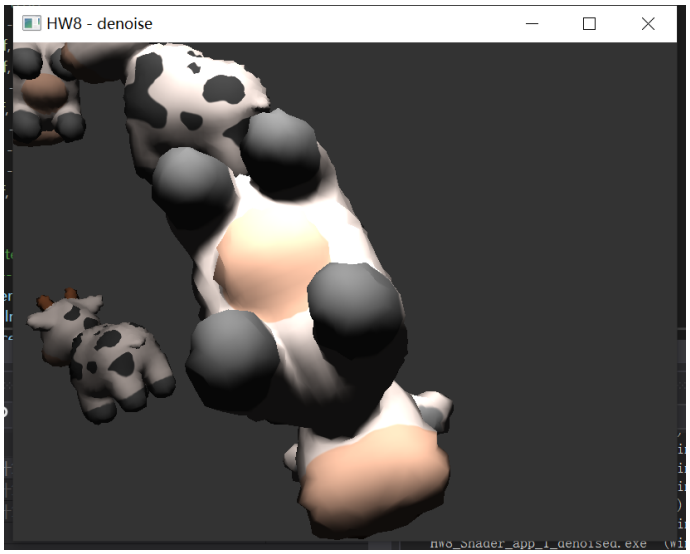

Figure 5.6: 带噪声的模型

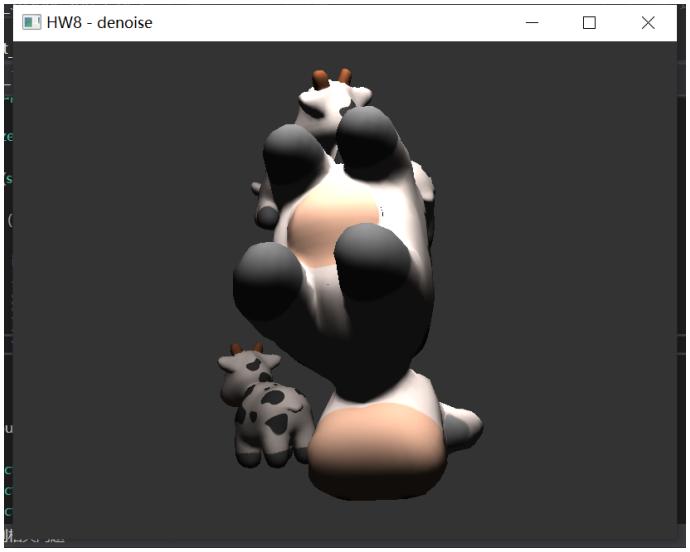

Figure 5.7: 降噪后的模型