

1. UV 错位：

scanline 的时候一开始写的 $\text{right.x} - \text{left.x}$ 求出 line 的最右端减最左端的一个单位平均步长值 step，后面从左到右插值的时候 $\text{left} + i \cdot \text{step}$

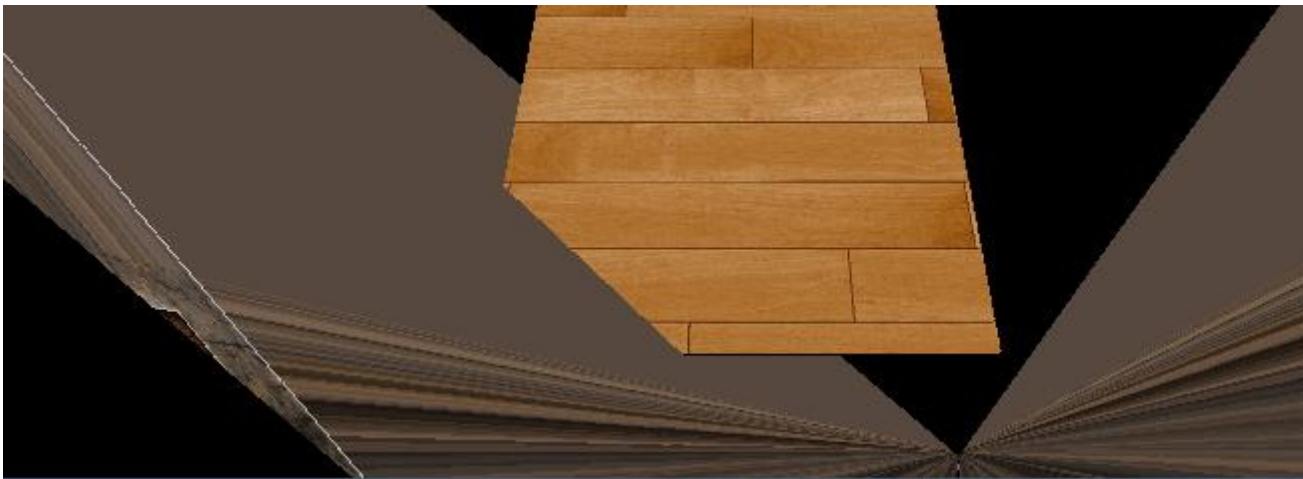
因为 right.x 和 left.x 都是 float，后面由可能计算的是 1.3 到 10.1 之间的单位步长，但是填入屏幕像素的时候是填整数 2 到 11 的位置，所以这个小数点误差造成了错位。先把 left 和 right 插值到 2.0 和 11.0 再计算单位步长 step 或者每次 $i++$ 的时候重新计算新的 left 插值都可以。

2. 纹理边缘会有彩色点



计算 uv 位置的时候直接 $v \cdot \text{uv_size} + u$ ，因为比如此时纹理是 256 的贴图，其实再内存里取值应该是 0-255. 直接用 256 就多出来一位，取到超出内存块最后的值，所以应该用 $\text{uv_max_size} = \text{uv_size} - 1$ 来给上面 v 乘。

3. 相机和观察对象移到很近的时候会把对象拉花



因为相机和观察很近的时候 $\text{mvp} \cdot v$ 得到的 w 分量是小于零的数，此时 w 已经失真了。

\therefore view 坐标系是先移动到世界坐标原点，再旋转朝向到指定角度；

又 $\therefore \text{ject} \cdot P(x, y, z, 1) \Rightarrow p'(x', y', z', w)$, $x' = d \cdot x / z$; $y' = d \cdot y / z$; $z' = (a \cdot z + b) / z$, $w = z$

\therefore 这个 w 是原先透视投影前的 z。当 $w < 0.0f$ 的时候其实就是在相机后面了。

(这个最坑，一直以为是裁剪写得不对)。

4. 正反面剔除写在归一化之前，导致法线不对。正反面剔除应该在归一化之后。

Pipeline_Process:

1. `model = WVP * V;`
2. `normalization(){postion/w, uv/w, color/w}`
3. `backface_culling();`
4. `clip(){split_triangle()}`
5. `toscreen();`
6. `scanline(){postion, uv*w, color*w}`