T4M

背景

这是多贴图混合的地表shader。在原版T4M的基础上，将surface shader替换为vertex-pixel shader，便于修改(例如自定义雾效)。删除了一些不必要的变体(defered pass)。

参数说明

1. 纹理使用数目。使用几张下拉框勾选几张。越多性能越差
2. 使用类型：1. 法线 2. 法线高光 3. 仅漫反射 4. 高光
3. SpecularColor ： 高光颜色
4. LayerXShininess：X层的光泽度
5. LayerX：X层的贴图
6. LayerXNormalmap：X层的法线贴图
7. Normal Factor X: X层的法线强度
8. Control ： 区域控制贴图，RGBA共四层
9. 潮湿：见场景shader
10. 高度雾：见场景shader
11. 阴影颜色

技术说明

1. 核心代码

```
    // 贴图混合计算
        void surf (Input IN, inout SurfaceOutput o) {
         // 控制遮罩，多通道，最多四通道
         half4 splat_control = tex2D (_Control, IN.uv_Control);
         half3 col;
         // 纹理贴图0~3
         half4 splat0 = tex2D (_Splat0, IN.uv_Splat0);
         half4 splat1 = tex2D (_Splat1, IN.uv_Splat1);
         #if defined(_TEXTURE_THREE) || defined(_TEXTURE_FOUR)
             half4 splat2 = tex2D (_Splat2, IN.uv_Splat2);
             #if defined(_TEXTURE_FOUR)
                 half4 splat3 = tex2D (_Splat3, IN.uv_Splat3);
             #endif
         #endif

         #if defined(_TYPE_BUMP) || defined(_TYPE_BUMPSPEC)
             float3 normalFactor0 = float3(_NormalFactor0, _NormalFactor0, 1);
             float3 normalFactor1 = float3(_NormalFactor1, _NormalFactor1, 1);
             float3 normalFactor2 = float3(_NormalFactor2, _NormalFactor2, 1);
             float3 normalFactor3 = float3(_NormalFactor3, _NormalFactor3, 1);
         #endif
         // Layer0
         col  = splat_control.r * splat0.rgb;
         // Layer0法线
         #if defined(_TYPE_BUMP) || defined(_TYPE_BUMPSPEC)
             o.Normal = splat_control.r * (UnpackNormal(tex2D(_BumpSplat0,
IN.uv_Splat0))*normalFactor0);
```

```
        #endif
        // Layer0 光滑度 高光强度
        #if defined(_TYPE_BUMPSPEC) || defined(_TYPE_SPECULAR)
            o.Gloss += splat0.a * splat_control.r ;
            o.Specular += _ShininessL0 * splat_control.r;
        #endif

        // Layer1
        col += splat_control.g * splat1.rgb;
        #if defined(_TYPE_BUMP) || defined(_TYPE_BUMPSPEC)
            o.Normal += splat_control.g * (UnpackNormal(tex2D(_BumpSplat1,
IN.uv_Splat1))*normalFactor1);
        #endif
        #if defined(_TYPE_BUMPSPEC) || defined(_TYPE_SPECULAR)
            o.Gloss += splat1.a * splat_control.g ;
            o.Specular += _ShininessL1 * splat_control.g;
        #endif

        // Layer2
        #if defined(_TEXTURE_THREE) || defined(_TEXTURE_FOUR)
            col += splat_control.b * splat2.rgb;
            #if defined(_TYPE_BUMP) || defined(_TYPE_BUMPSPEC)
                o.Normal += splat_control.b * (UnpackNormal(tex2D(_BumpSplat2,
IN.uv_Splat2))*normalFactor2);
            #endif
            #if defined(_TYPE_BUMPSPEC) || defined(_TYPE_SPECULAR)
                o.Gloss += splat2.a * splat_control.b ;
                o.Specular += _ShininessL2 * splat_control.b;
            #endif

            // Layer3
            #if defined(_TEXTURE_FOUR)
                col += splat_control.a * splat3.rgb;
                #if defined(_TYPE_BUMP) || defined(_TYPE_BUMPSPEC)
                    o.Normal += splat_control.a *
(UnpackNormal(tex2D(_BumpSplat3, IN.uv_Splat3))*normalFactor3);
                #endif
                #if defined(_TYPE_BUMPSPEC) || defined(_TYPE_SPECULAR)
                    o.Gloss += splat3.a * splat_control.a ;
                    o.Specular += _ShininessL3 * splat_control.a;
                #endif
            #endif
        #endif

        o.Specular = max(o.Specular, 0.001);
        o.Emission = col * _EmissionPower;
        o.Albedo = col;
        o.Alpha = 0.0;
    }
```

2. surface 转换相关
    1. 三个pass，用宏控制不同代码，合并重复部分
        1. forwardbase 前向光照基础pass
        2. forwardadd 额外灯光pass
        3. meta 为烘培设置固有色和自发光