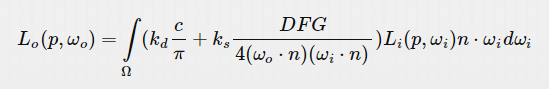
**Specular IBL**

在上一教程中，我们通过预先计算辐照度图作为照明的间接漫反射部分，将PBR与基于图像的光照结合起来。 在本教程中，我们将重点介绍反射率方程的镜面反射部分：

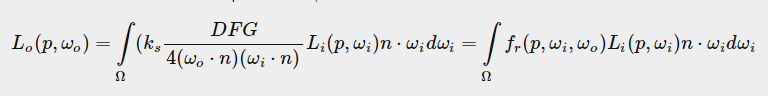


你会注意到，Cook-Torrance镜面反射部分（乘以kS）在积分上不是恒定的，并且取决于入射光的方向，还取决于视线的方向。 试图求解所有入射光方向（包括所有可能的视线方向）的积分情况组合过多，并且开销过大，无法实时计算。 Epic Games提出了一种解决方案，在考虑到一些折衷的情况下，他们能够实时对高光部分进行预卷积，这称为

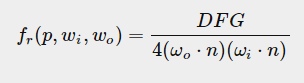
**split sum approximation**（分割和近似）。

分割总和近似将反射率方程式的镜面部分拆分为两个单独的部分，我们可以分别对其进行卷积，然后在PBR着色器中进行组合，以实现基于镜面间接图像的光照。 与我们预先对辐照度图进行卷积的方式类似，分割总和近似值需要将HDR环境图作为其卷积输入。

为了理解，我们将再次查看反射率方程，但这一次仅关注镜面反射部分（我们在上一教程中已提取了漫反射部分）：

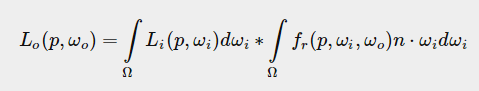


出于与辐照度卷积相同的（性能）原因，我们无法实时计算镜面反射部分的积分而同时获得合理的性能。 因此，最好我们预先计算该积分，以获得类似镜面反射的IBL贴图，使用片段的法线向量对该贴图进行采样并完成此操作。 但是，这有点棘手。 我们可以预先计算辐照度图，因为积分仅取决于ωi，并且可以将常数扩散反照率项移出积分。 这次，积分不仅取决于ωi，从BRDF中可以看出：



这积分也取决于ωo，我们无法真正对具有两个方向向量的预先计算的立方体贴图进行采样。 位置p与上一教程中所述的位置无关。 在实时设置中，无法为ωi和ωo的每种可能组合预先计算该积分。

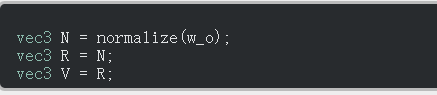
Epic Games的拆分总和近似值，通过将预计算分为两个单独的部分来解决此问题，我们稍后可以将这些部分合并以获得我们想要的最终预计算结果。 分割和近似将镜面反射积分拆分为两个单独的积分：



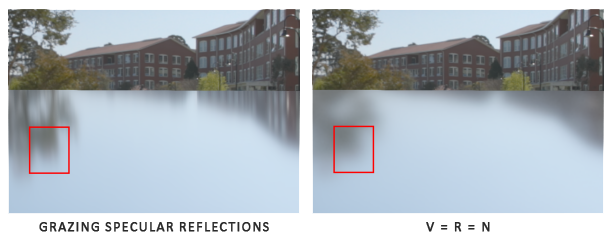
第一部分（卷积时）称为预过滤环境图，该图（类似于辐照度图）是预计算的环境卷积图，但是这次考虑了粗糙度。 为了增加粗糙度，将环境图与更多分散的样本向量进行卷积，从而产生更多的模糊反射。 对于每个卷积的粗糙度级别，我们将顺序模糊的结果存储在预先过滤的贴图的mipmap级别中。 例如，一个预先过滤的环境图在其5个mipmap级别中存储了5个不同粗糙度值的预卷积结果，如下所示：



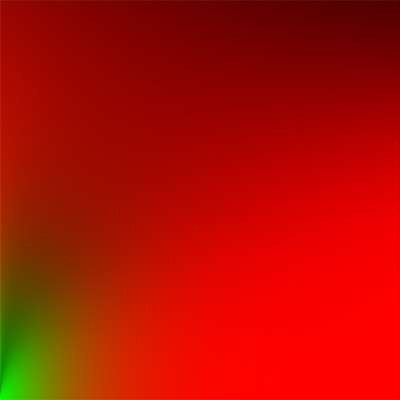
我们使用Cook-Torrance BRDF的正态分布函数（NDF）生成样本矢量及其散射强度，该函数以法线方向和视线方向为输入。 正如我们在对环境图进行卷积时事先不知道视图方向，Epic Games通过假设视线方向（同镜面反射方向）始终等于输出采样方向ωo来进行进一步近似。 这会将其自身转换为以下代码：



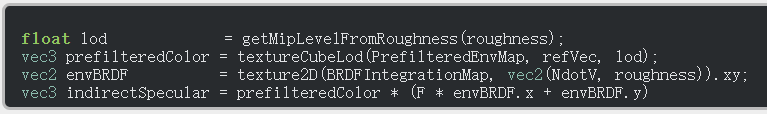
这样，预过滤的环境卷积不需要知道视线方向。 这的确意味着当从下图的角度观察镜面反射时，我们没有得到很好的掠光镜面反射；但是，通常认为这是一个不错的折衷方式：



等式的第二部分等于镜面积分的BRDF部分。 如果我们假设每个方向的入射辐射完全是白色的（因此L（p，x）= 1.0），则可以在给定输入粗糙度和法线n与光方向ωi，或n⋅ωi的输入角度的情况下预先计算BRDF的响应。 Epic Games在称为BRDF集成贴图的2D查找纹理（LUT）中将变化后的粗糙度值上预先计算的BRDF对每个法线和光照方向组合的响应存储起来。 2D查找纹理向表面的菲涅尔效应输出比例（红色）和偏差值（绿色），从而为我们提供了镜面积分的第二部分：



我们通过将平面的水平纹理坐标（介于0.0和1.0之间）视为BRDF的输入n⋅ωi，并将其垂直纹理坐标作为输入粗糙度值来生成查找纹理。 借助此BRDF集成贴图和预过滤的环境图，我们可以将两者结合起来以获得镜面积分的结果：

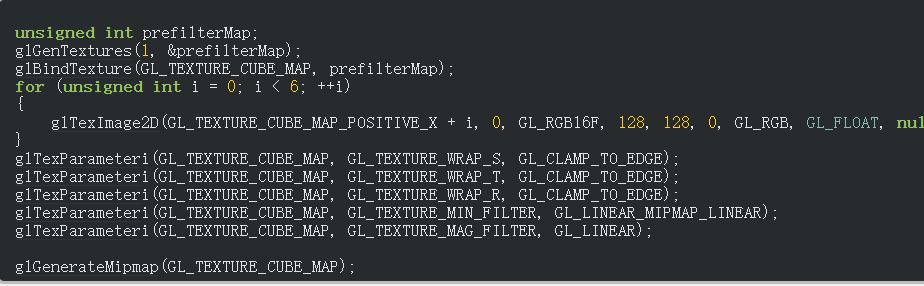


这应该让你大致了解Epic Games的分割和近似如何大致接近反射率方程式的间接镜面反射部分。 现在，让我们尝试自己构建预卷积的部分。

**Pre-filtering an HDR environment map**

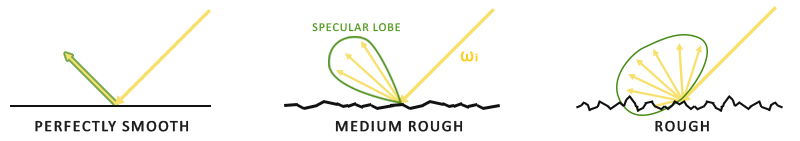
预过滤环境图与我们对辐照度图进行卷积非常相似。 不同之处在于，我们现在考虑了粗糙度，并在预先过滤的贴图的mip级别中依次存储了逐渐粗糙的反射。

首先，我们需要生成一个新的立方体贴图来保存预先过滤的环境贴图数据。 为确保为其Mip级别分配足够的内存，我们将**glGenerateMipmap**称为分配所需内存量的简便方法。



请注意，由于我们计划对**prefilterMap**的mipmap进行采样，因此您需要确保将其最小化过滤器设置为GL\_LINEAR\_MIPMAP\_LINEAR以启用三线性过滤。 我们以其基点水平以128 x 128的每面分辨率存储预先过滤的镜面反射。 对于大多数反射来说，这可能就足够了，但是如果您有大量光滑的材质（例如汽车的反射），则可能需要提高分辨率。

在上一个教程中，我们通过使用球坐标生成均匀分布在半球Ω上的样本矢量来对环境图进行卷积。 尽管这对于辐照效果很好，但对于镜面反射效率却较低。 当涉及镜面反射时，基于表面的粗糙度，光在法线n上围绕反射向量r紧密或大致反射，但是（除非表面非常粗糙）在反射向量附近：



可能的出射光反射的一般形状称为镜面叶。 随着粗糙度的增加，镜面叶的大小增加； 镜面叶的形状随入射光方向的变化而变化。 因此，镜面叶的形状对材质有着高度的依赖。

当涉及到微表面模型时，我们可以将镜面叶想象为在给定某些入射光方向的情况下，围绕微面中途矢量的反射方向。 看到大多数光线最终终止在微面中途矢量周围反射的镜面叶中，因此以与大多数情况下会被浪费的相似方式生成样本矢量是有意义的。 此过程称为**importance sampling**（重要性抽样）。

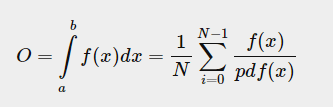
### Monte Carlo integration and importance sampling

为了充分了解重要性抽样的重要性，我们首先深入研究称为蒙特卡洛积分的数学结构。 蒙特卡洛积分主要围绕统计学和概率论的结合进行。 蒙特卡洛（Monte Carlo）帮助我们离散地解决了计算总体的某些统计量或价值的问题，而不必考虑所有总体。

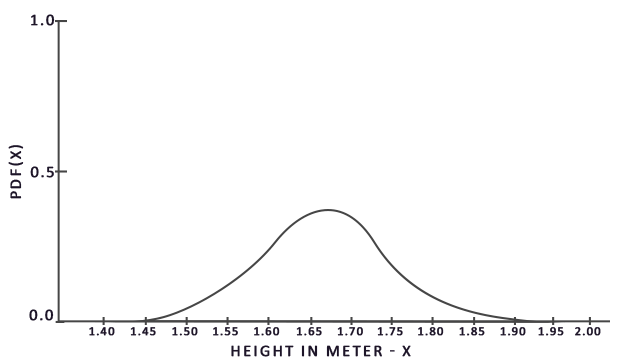
例如，假设要计算一个国家所有公民的平均身高。为了获得结果，您可以测量每个公民的身高，并平均其身高，这将为您提供所需的确切答案。但是，由于大多数国家/地区都有大量人口，所以这不是现实的方法：这将花费过多的精力和时间。

一种不同的方法是从总体中选择更小的完全随机（无偏）子集，测量其高度并将结果平均。这个人口可能只有100人。虽然不如确切答案准确，但您会得到一个相对接近基本事实的答案。这被称为大数定律。这个想法是，如果从总人口中测量较小的一组真正随机样本的大小N，结果将相对接近真实答案，并且随着样本数量N的增加而变得更接近。

蒙特卡洛积分建立在大数定律的基础上，并采用相同的方法求解积分。无需求解所有可能（理论上无限）样本值x的积分，只需生成从总人口和平均值中随机选择的N个样本值即可。随着N的增加，我们保证得到的结果更接近于积分的确切答案：



为了求解积分，我们对总体a到b取N个随机样本，将它们加在一起，然后除以样本总数，以求平均值。 pdf代表概率密度函数，该函数告诉我们特定样本在整个样本集中发生的概率。 例如，人口高度的pdf看起来像这样：



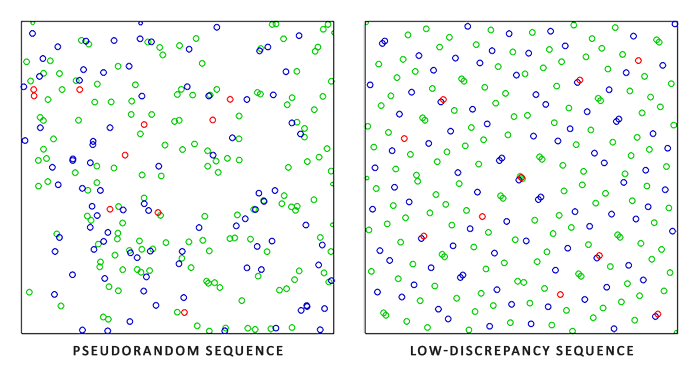
从这张图中我们可以看到，如果我们随机抽取总体样本，则选择身高为1.70的某人的机会较高，而该身高为1.50的概率较低。

当涉及蒙特卡洛积分时，某些样本可能比其他样本具有更高的生成概率。这就是为什么对于任何一般的蒙特卡洛估计，我们都会根据pdf将采样值除以或乘以采样概率。到目前为止，在我们估计积分的每种情况下，我们生成的样本都是统一的，生成的机会完全相同。到目前为止，我们的估计是无偏的，这意味着，随着样本数量的不断增加，我们最终将收敛到积分的精确解。

但是，某些蒙特卡洛估计量是有偏差的，这意味着生成的样本并不是完全随机的，而是集中于特定的值或方向。这些有偏的蒙特卡洛估计量具有更快的收敛速度，这意味着它们可以更快的速度收敛到精确解，但是由于有偏的性质，很可能他们永远不会收敛到精确解。通常，这是可以接受的折衷方案，尤其是在计算机图形学中，因为只要结果在视觉上可以接受，确切的解决方案就不太重要。正如我们很快就会看到的重要性采样（使用有偏的估计器）一样，生成的样本偏向特定的方向，在这种情况下，我们通过将每个样本乘以或除以相应的pdf来解决这个问题。

蒙特卡洛积分在计算机图形学中非常普遍，因为它是一种以离散有效的方式近似连续积分的直观方法：对任何面积/体积进行采样（例如半球Ω），在该面积内生成N个随机样本/ volume，求和并权衡每个样本对最终结果的贡献。

蒙特卡洛积分是一个广泛的数学主题，我将不进一步深入研究其具体细节，但是我们将提到，还有多种生成随机样本的方法。默认情况下，每个样本都是我们习惯的完全（伪）随机样本，但是通过利用半随机序列的某些属性，我们可以生成仍然是随机样本但具有有趣性质的样本向量。例如，我们可以对称为低差异序列的东西进行蒙特卡洛积分，该序列仍会生成随机样本，但每个样本分布得更均匀：



当使用低差异序列生成蒙特卡洛样本矢量时，该过程称为准蒙特卡洛积分。准蒙特卡罗方法具有更快的收敛速度，这使它们对于性能要求很高的应用很有趣。

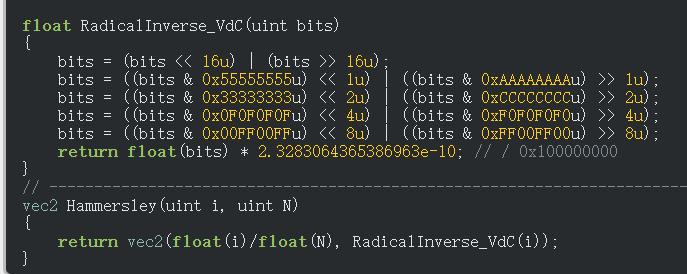
鉴于我们新获得的有关蒙特卡洛（Monte Carlo）和拟蒙特卡洛（Quasi-Monte Carlo）积分的知识，我们可以使用一个有趣的属性来获得更快的收敛速度，即重要性抽样。我们在本教程中之前已经提到过，但是对于光的镜面反射，反射的光矢量被限制在镜面波瓣中，其大小取决于表面的粗糙度。看到镜面叶外的任何（准）随机生成的样本与镜面积分无关，因此将样本生成集中在镜面波叶是有意义的，但要付出使蒙特卡洛估计量产生偏差的代价。

本质上，这就是重要性的重要性所在：在某些区域生成样本矢量，该区域受围绕微面中途矢量的粗糙度限制。通过将准蒙特卡洛采样与低差异序列相结合，并使用重要性采样对样本矢量进行偏置，我们可以获得很高的收敛速度。因为我们以更快的速度获得解决方案，所以我们需要更少的样本才能达到足够的近似值。因此，该组合甚至允许图形应用程序实时求解镜面积分，尽管它比预计算结果要慢得多。

### A low-discrepancy sequence

在本教程中，我们将使用重要性采样来预先计算间接反射方程的镜面反射部分，该采样基于Quasi-Monte Carlo方法给出了随机的低差异序列。 正如Holger Dammertz精心描述的那样，我们将使用的序列称为Hammersley序列。 Hammersley序列基于Van Der Corpus序列，该序列在其小数点附近镜像了一个十进制的二进制表示形式。

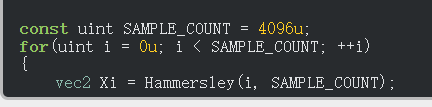
给出一些巧妙的技巧，我们可以在着色器程序中非常有效地生成Van Der Corpus序列，我们将使用它来获得N个样本总数中的Hammersley序列样本i：



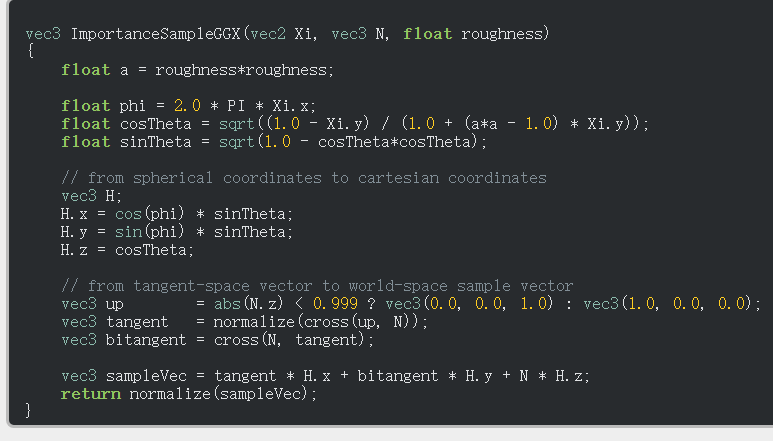
GLSL Hammersley函数为我们提供了大小为N的总样本集中的低差异样本i。

### GGX Importance sampling

代替在积分半球Ω上均匀或随机地（蒙特卡罗）生成样本矢量，我们将根据表面的粗糙度生成偏向微表面中途矢量的一般反射方向的样本矢量。 采样过程将与我们之前看到的过程相似：开始一个大循环，生成一个随机（低差异）序列值，将该序列值用于在切线空间中生成样本矢量，变换到世界空间并对场景的辐照度进行采样。 不同之处在于，我们现在使用低差异序列值作为输入来生成样本矢量：

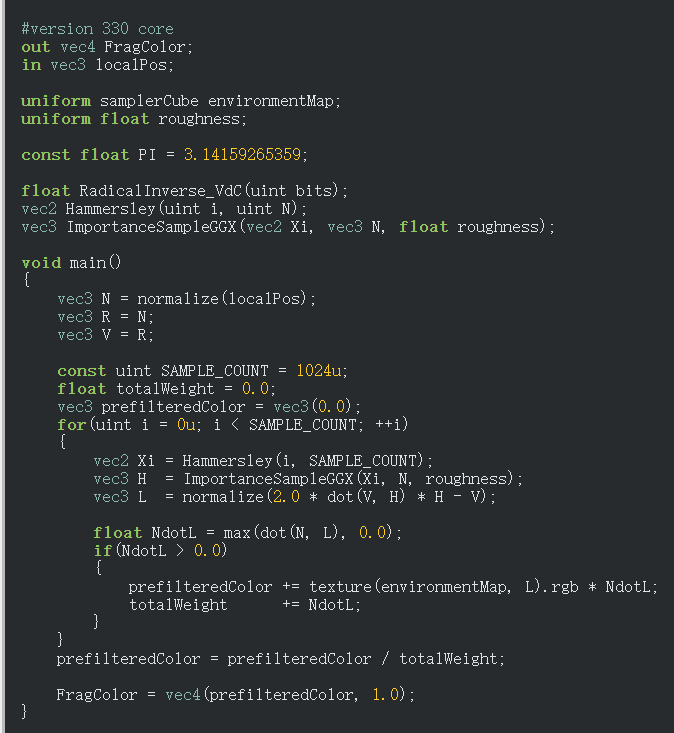


此外，要构建样本向量，我们需要一些方法来将样本向量定向和偏向某些表面粗糙度的镜面叶。 我们可以按照Theory教程中所述使用NDF，并将GGX NDF结合到Epic Games所述的球形样本矢量处理中：



基于一些输入粗糙度和低差异序列值Xi，这为我们提供了一个样本矢量，该矢量略微围绕预期的微表面的中途矢量。 请注意，根据迪士尼最初的PBR研究，Epic Games使用平方粗糙度来获得更好的视觉效果。

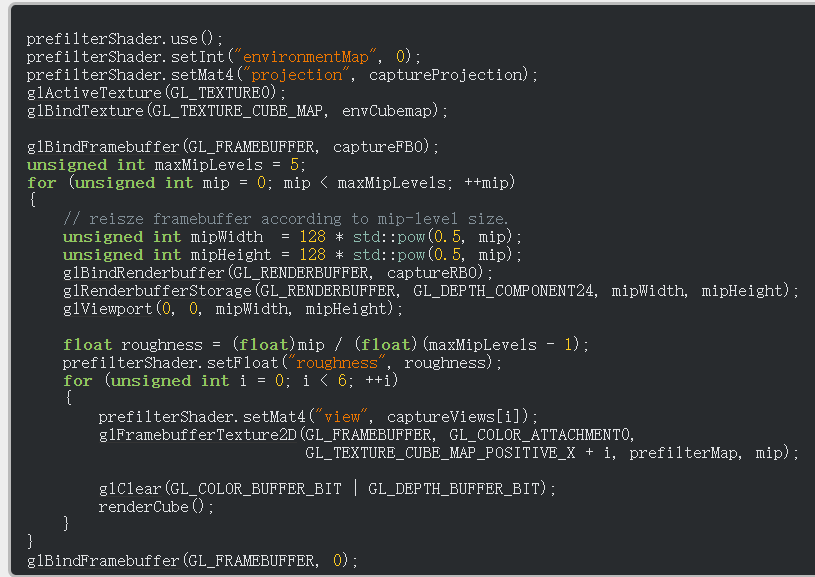
使用低差异Hammersley序列和定义的样本生成，我们可以最终确定预滤波卷积着色器：



我们根据在预过滤器立方体贴图的每个mipmap级别（从0.0到1.0）上变化的一些输入粗糙度，对环境进行预过滤，并将结果存储在prefilteredColor中。 所得的prefilteredColor除以总样本权重，其中对最终结果影响较小的样本（对于较小的NdotL）对最终权重的贡献较小。

### Capturing pre-filter mipmap levels

剩下要做的就是让OpenGL在多个mipmap级别上根据不同的粗糙度值预过滤环境贴图。 使用辐照度教程的原始设置实际上很容易做到：

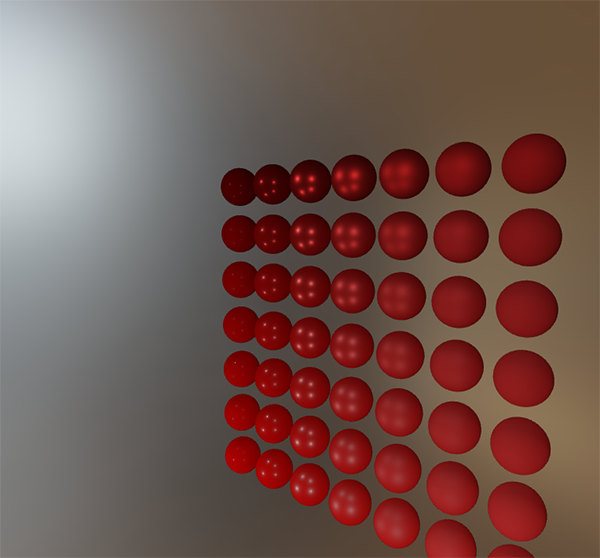


此过程类似于辐照度贴图卷积，但是这次我们将帧缓冲区的尺寸缩放到适当的mipmap比例，每个mip级别将尺寸减小2。此外，我们在glFramebufferTexture2D的最后一个参数中指定要渲染到的mip级别， 将我们要预过滤的粗糙度传递给预过滤器着色器。

这将为我们提供一个经过适当预过滤的环境贴图，当我们从中访问该贴图时，它会返回模糊的反射。 如果我们在天空盒着色器中显示预先过滤的环境立方体贴图，并在其着色器中强制在其第一个Mip级别之上采样，如下所示：



我们得到的结果确实看起来像原始环境的模糊版本：



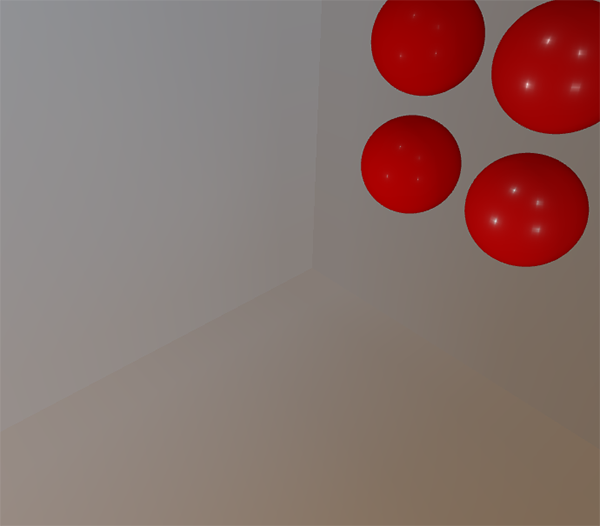
如果看起来有些相似，则说明您已经成功预过滤了HDR环境图。 尝试不同的Mipmap级别，以查看预过滤器贴图在增加的Mip级别上从锐利反射逐渐变为模糊反射。

## Pre-filter convolution artifacts

尽管当前的预过滤器贴图可以在大多数情况下正常工作，但迟早您会遇到几个与预过滤器卷积直接相关的渲染工件。 我将在这里列出最常见的方法，包括如何修复它们。

### Cubemap seams at high roughness

在具有粗糙表面的表面上对预过滤贴图进行采样意味着在一些较低的Mip级别上对预过滤贴图进行采样。 采样立方体贴图时，默认情况下，OpenGL不会在立方体贴图的各个面之间进行线性插值。 由于较低的Mip级别都具有较低的分辨率，并且预过滤图被更大的样本叶所盘绕，因此立方面之间的缺乏过滤变得非常明显：



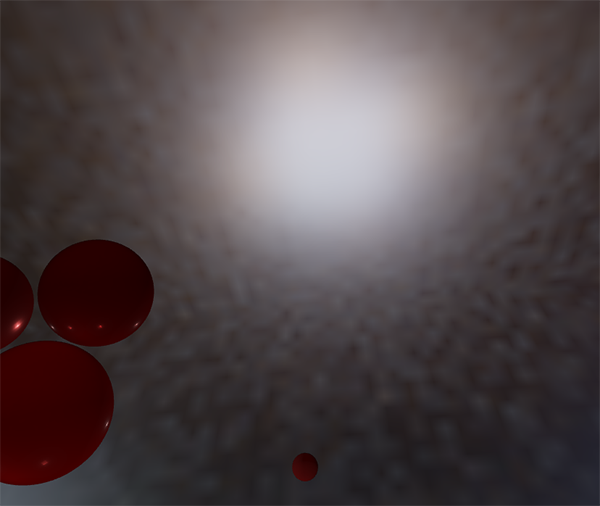
幸运的是，OpenGL通过启用以下选项为我们提供了在立方体之间正确过滤的选项：



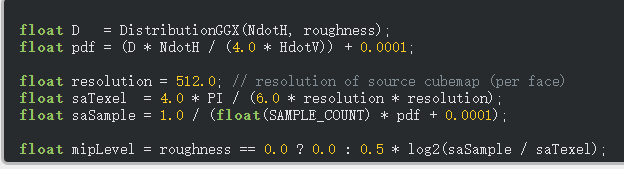
只需在应用程序开始的某个地方启用此属性，接缝就会消失。

### Bright dots in the pre-filter convolution

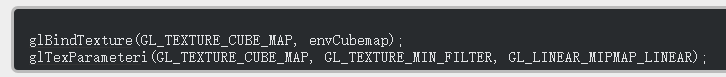
由于高频细节和镜面反射的光强度的变化很大，对镜面反射进行卷积需要大量样本才能正确对应HDR环境反射的变化性质。 我们已经进行了大量的采样，但是在某些环境下，在某些较粗糙的mip级别上可能仍然不够，在这种情况下，您将开始看到在明亮区域周围出现点状图案：



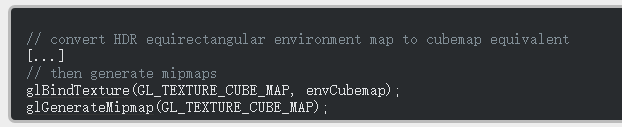
一种选择是进一步增加样本数量，但这还不足以适用于所有环境。 如Chetan Jags所述，我们可以通过（在预过滤卷积期间）不直接采样环境图，而是根据积分的PDF和粗糙度采样环境图的mip级别来减少此伪像：



不要忘记在要从以下位置采样其Mip级别的环境地图上启用三线性过滤：



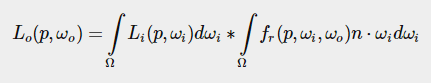
设置立方体贴图的基本纹理后，让OpenGL生成mipmap：



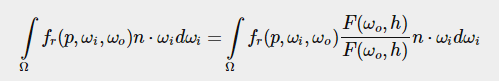
这出奇地好，并且可以去除预过滤贴图中较粗糙表面上的大多数（如果不是全部）点。

## Pre-computing the BRDF

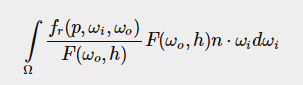
在预过滤的环境准备并运行后，我们可以集中精力于求和近似的第二部分：BRDF。 让我们再次简要回顾一下镜面反射拆分求和近似：

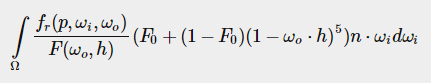


我们已经在不同粗糙度级别上预先计算了预过滤器贴图中的分割总和近似的左部分。 右边部分要求我们在角度n⋅ωo，表面粗糙度和菲涅耳F0上对BRDF方程进行卷积。 这类似于将镜面BRDF与纯白色环境或恒定辐射Li为1.0集成在一起。 在3个变量上对BRDF进行卷积有点很多，但是我们可以将F0从镜面BRDF方程中移出：

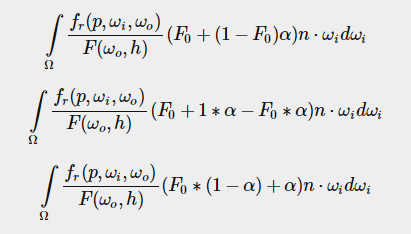


F是菲涅耳方程。 将菲涅耳分母移到BRDF可以得到以下等式：

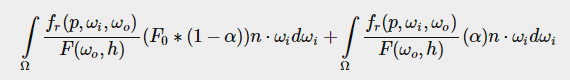
  
用最接近的F替换为Fresnel-Schlick近似值，我们得到：



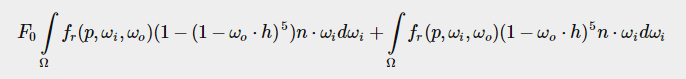
让我们将（1-ωo⋅h）5替换为α，更轻松地求解F0：



然后，我们将菲涅耳函数F分成两个积分：



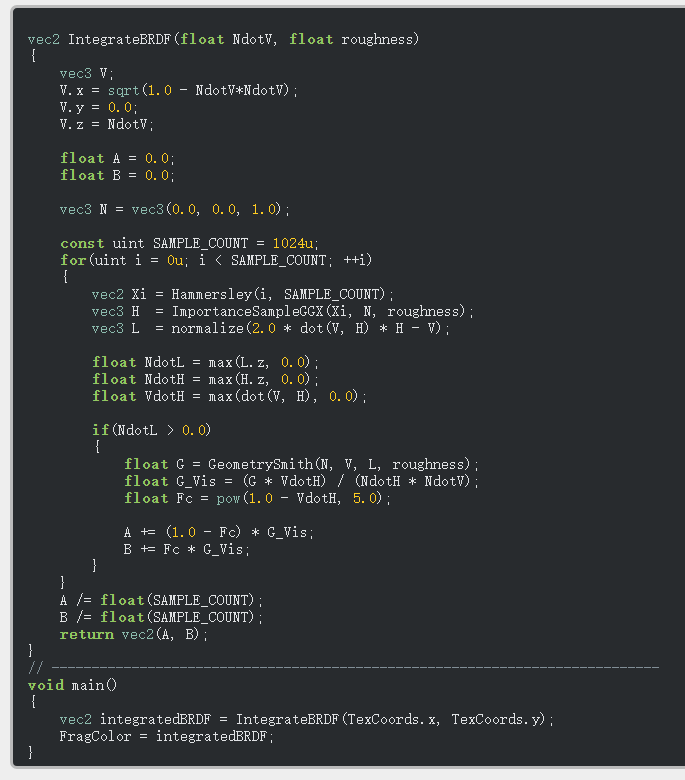
这样，F0在积分上是恒定的，我们可以从积分中取出F0。 接下来，我们将α替换为其原始形式，从而得到最终的分和BRDF方程：



两个结果积分分别表示比例和对F0的偏差。 请注意，由于f（p，ωi，ωo）已经包含F项，因此它们都抵消了，从而从f中删除了F。

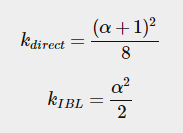
与早期的卷积环境贴图相似，我们可以在它们的输入上对BRDF方程进行卷积：n和ωo之间的角度以及粗糙度，然后将卷积的结果存储在纹理中。 我们将卷积后的结果存储在称为BRDF集成图的2D查找纹理（LUT）中，稍后将其用于PBR片段着色器中，以获得最终的卷积间接镜面反射结果。

BRDF卷积着色器将2D纹理坐标直接用作BRDF卷积的输入（NdotV和粗糙度），从而在2D平面上运行。 卷积代码与前置滤波器卷积在很大程度上相似，不同之处在于，它现在根据我们的BRDF的几何函数和Fresnel-Schlick的近似来处理样本矢量：

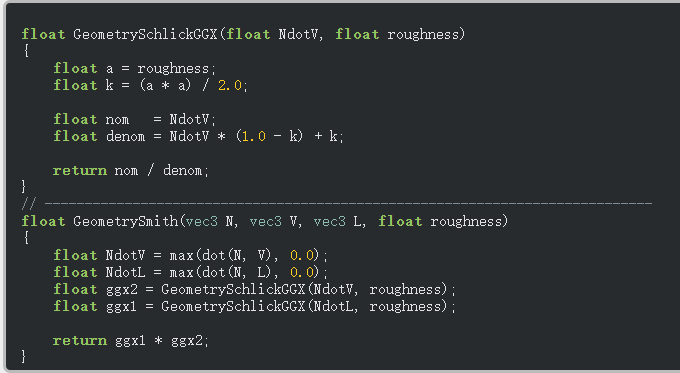


如你所见，BRDF卷积是从数学到代码的直接转换。 我们将角度θ和粗糙度作为输入，生成具有重要度采样的样本矢量，在几何形状和BRDF的菲涅耳项上进行处理，并为每个样本输出比例和对F0的偏差，最后将其平均。

你可能已经从理论教程中回想起，与IBL一起使用时，BRDF的几何术语略有不同，因为其k变量的解释略有不同：

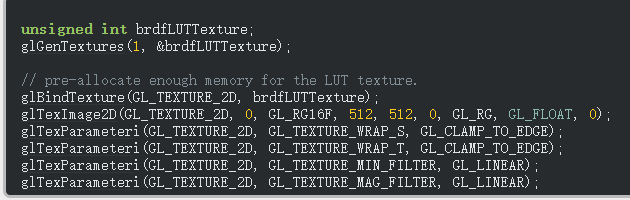


由于BRDF卷积是镜面IBL积分的一部分，因此我们将kIBL用于Schlick-GGX几何函数：



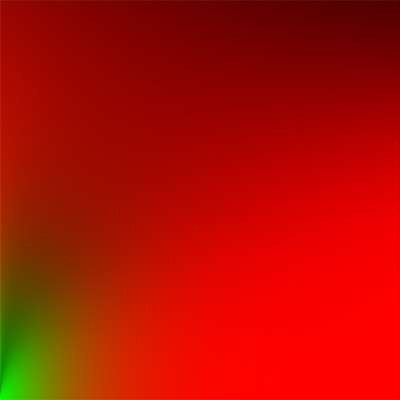
注意，虽然k以a为参数，但我们并没有像a最初对a的其他解释那样对粗糙度进行平方。 可能已经在这里平方了。 我不确定这是否与Epic Games的部分或迪斯尼原始纸上的不一致，但直接将粗糙度转换为会得到与Epic Games版本相同的BRDF集成图。

最后，为了存储BRDF卷积结果，我们将生成512 x 512分辨率的2D纹理。



请注意，我们使用Epic Games建议的16位精度浮点数。 确保将环绕方式设置为GL\_CLAMP\_TO\_EDGE，以防止边缘采样伪像。

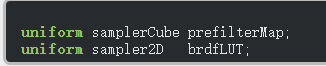
然后，我们重新使用相同的帧缓冲区对象，并在NDC屏幕空间四边形上运行此着色器：



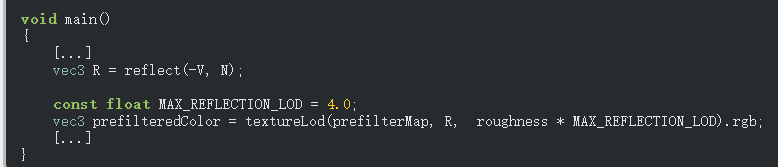
使用预过滤的环境贴图和BRDF 2D LUT，我们可以根据分割总和近似值重建间接镜面反射积分。 然后，合并的结果将充当间接或环境镜面光。

## Completing the IBL reflectance

为了使反射率方程式的间接镜面部分正常运行，我们需要将分离和近似的两个部分缝合在一起。 首先，将预先计算的光照数据添加到PBR着色器的顶部：

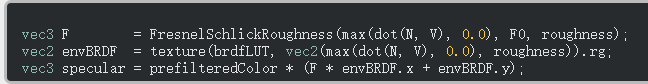


首先，通过使用反射向量对预过滤的环境贴图进行采样，可以获取表面的间接镜面反射。 请注意，我们根据表面粗糙度对适当的Mip级别进行采样，从而使较粗糙的表面模糊了镜面反射。



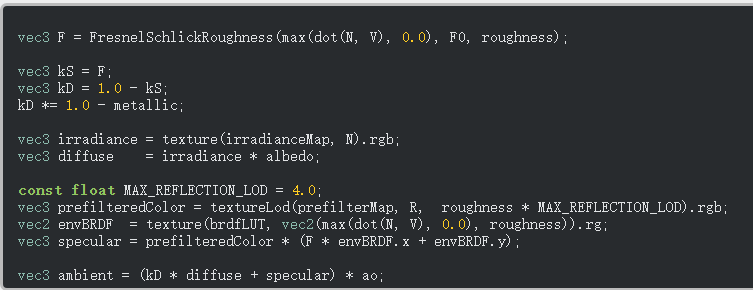
在预过滤器步骤中，我们最多仅将环境贴图卷积最多5个mip级别（0到4），此处将其表示为MAX\_REFLECTION\_LOD，以确保在没有（相关）数据的情况下不对mip级别进行采样。

然后，在给定材料的粗糙度以及法线和视图矢量之间的角度的情况下，从BRDF查找纹理中采样：



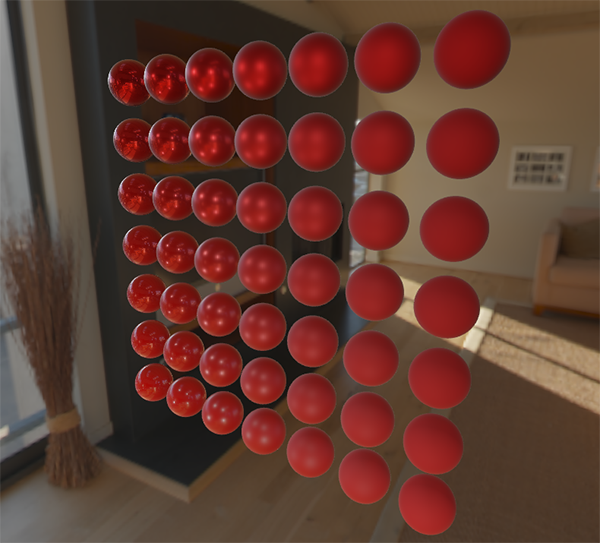
给定BRDF查找纹理的缩放比例和对F0的偏差（这里我们直接使用间接菲涅耳结果F），我们将其与IBL反射率方程的左侧预滤光片部分结合在一起，并将近似积分结果重构为镜面反射 。

这给了我们反射率方程式的间接镜面部分。 现在，将其与上一教程中的反射率方程的漫反射部分结合起来，我们将获得完整的PBR IBL结果：



请注意，我们没有将镜面反射乘以kS，因为那里已经有菲涅耳乘法。

现在，在一系列因其粗糙度和金属特性而异的球体上运行此确切代码，我们终于可以在最终的PBR渲染器中看到其真实颜色：



我们甚至可以crazy一点，使用一些很酷的纹理PBR材质：

