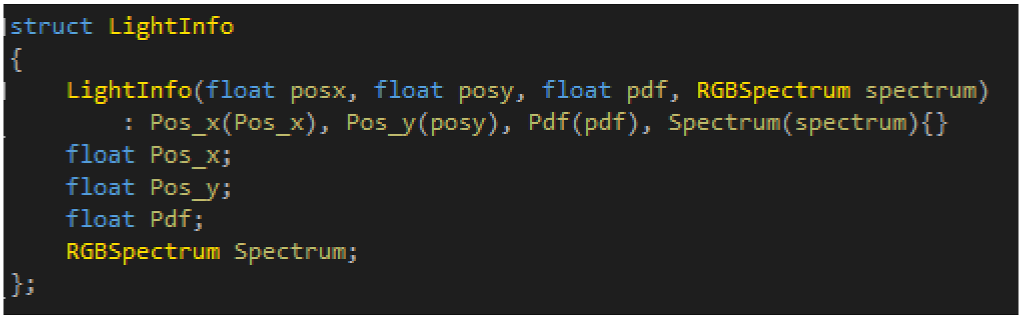
Rendering – HW3

1. methods

target: A light probe image is subdivided into 64 equal-energy regions. A point light is created for each region at its centroid.

First. I design a structure to store the useful information to Sample_L() named LightInfo.



```
struct LightInfo
{
    LightInfo(float posx, float posy, float pdf, RGBSpectrum spectrum)
        : Pos_x(Pos_x), Pos_y(posy), Pdf(pdf), Spectrum(spectrum){}
    float Pos_x;
    float Pos_y;
    float Pdf;
    RGBSpectrum Spectrum;
};
```

Pos_x, Pos_y means the center position of the region.

Pdf: use for sampled infinite light direction.

Spectrum: store the range color information of the region.

Then, We need to create the structure data in initial method and use the structure data in Sample_L()

In EnvironmentLight() initial method:

The basic input is the texture map, we need to use it to create the radiance map and scale it first.

Then, we need to create the integrator image to use as follows. The basic idea of the map is each of the cell store the energy from (0,0) to the cell. It convenient for us to calculate the energy of the region any of the image. The next show the core code in here.

itgimg[pos_now] = itgimg[pos_left] + itgimg[pos_up] - itgimg[pos_leftup] + img[pos_now] * SolidAngle;

The next is the core code of the median cut method:

For each split times:

> Judge split the region to two by equal energy. If the width is long than height, we search colume one by one. Otherwise, we search rows one by one.
>
> Each step, we calculate the now region’s energy by the integrator image generated above. It’s core idea is: itgimg(I + h - 1, j + w - 1) – itgimg(I + h - 1, j) – itgimg(I, j + w - 1) + itgimg(I – 1, j - 1);
>
> Finally, we only store the region this step generated.

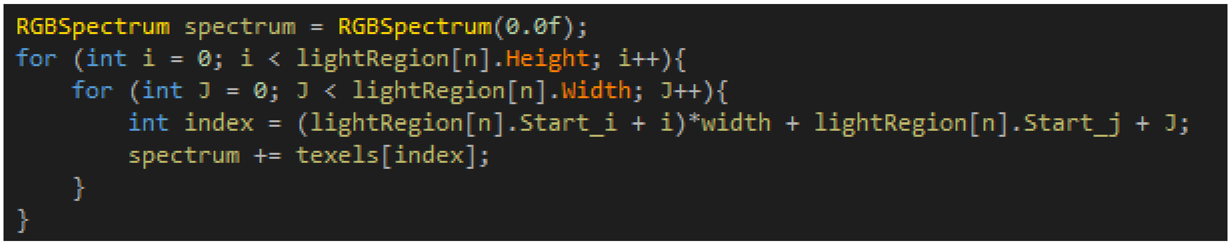
We define a structure LightRegion to store the information used in the generated process. It include the region’s left up point position, the region hight and width, the region energy.

We need to generate LightInfo structure data in the end. We need to calculate the region center position in this step.

Center = left up point(I, j) + (h, w) / 2;

But notice here we need to normalize the position of the image.

We also need to collect each region’s color informations in spectrum. You can see the follow pixture tp know how to do it.

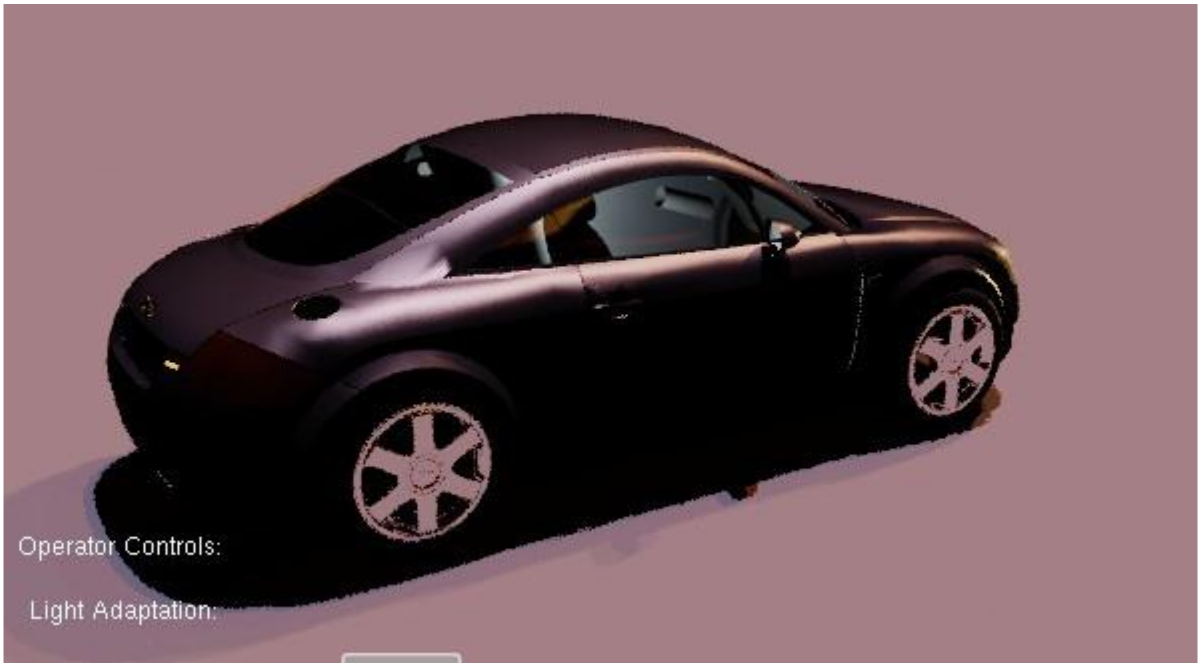


```
RGBSpectrum spectrum = RGBSpectrum(0.0f);
for (int i = 0; i < lightRegion[n].Height; i++){
    for (int J = 0; J < lightRegion[n].Width; J++){
        int index = (lightRegion[n].Start_i + i)*width + lightRegion[n].Start_j + J;
        spectrum += texels[index];
    }
}
```

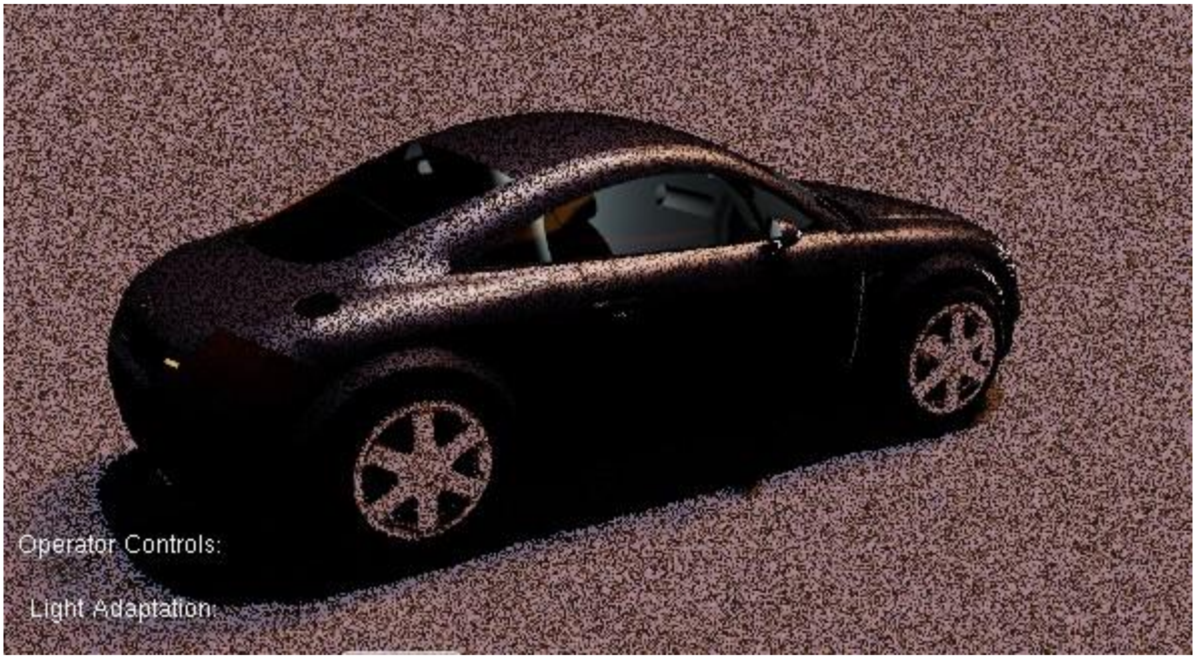
The pdf = 1 / lightRegion.size();

2. results

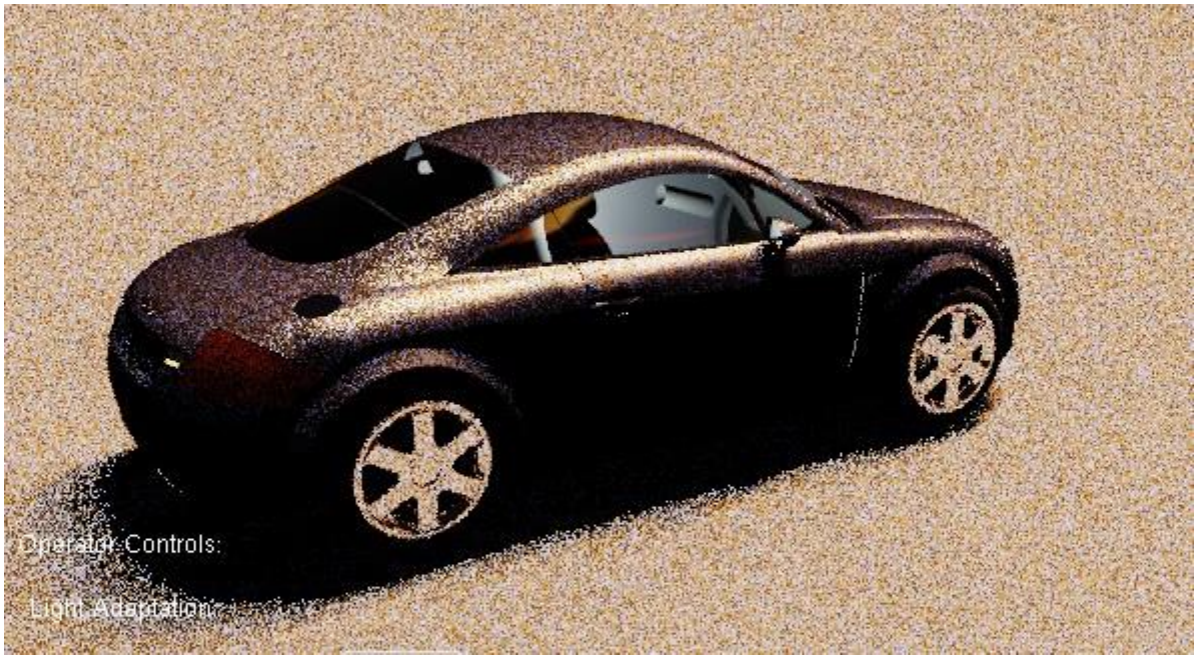
Split the image into 16 regions: [19.8 s]



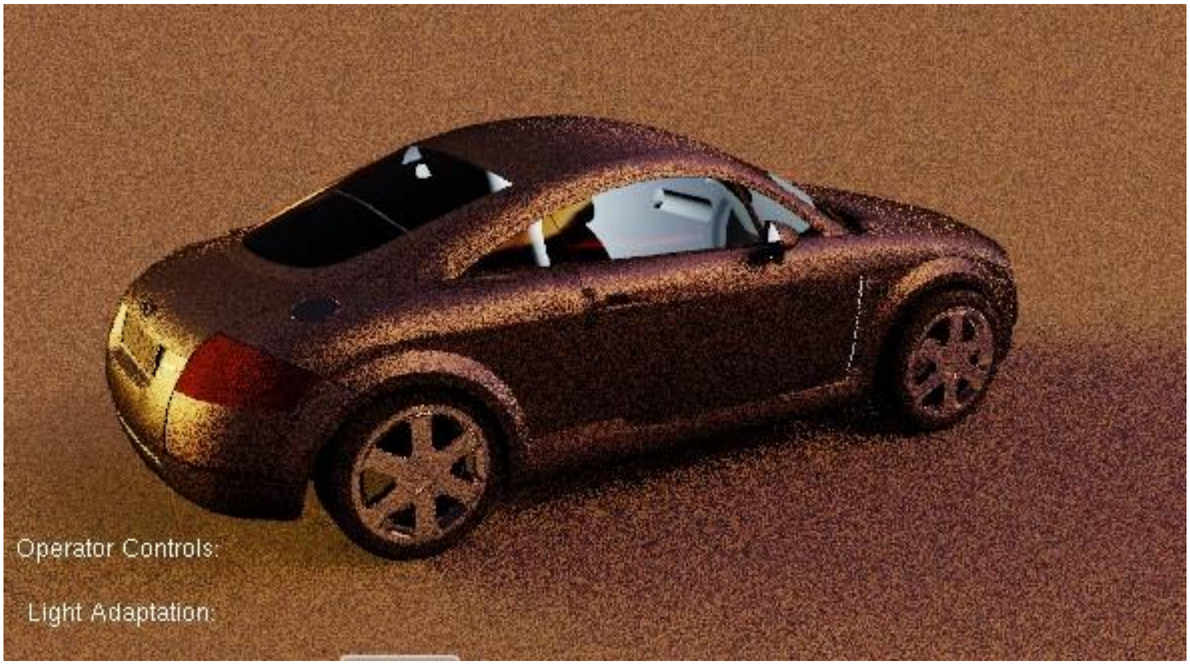
Split the image into 64 regions: [20.2 s]



Split the image into 256 regions: [20.0 s]



The infinite method in pbrt results:


Operator Controls:
Light Adaptation: