Embree – Multiple Importance Sampling

Luís Paulo Santos, April, 2018

This tutorial with `Embree` will modify a copy of the `viewer` tutorial's code:

1. Download the file `VI2_EmbreeT5_MIS.cpp` made available on the web site and copy it to `$EMBREE_SOURCES$/tutorials/viewer/`
2. Modify your `Visual Studio` solution or `Makefile` or even `CMake` file, such that the `viewer` project (included in the tutorials) compiles `VI2_EmbreeT5_MIS.cpp` instead of `viewer_device.cpp`

NOTE: Steps 3 and 4 below do not apply if you did this during Tutorial4. If not the files `materials.h` and `obj_loader.cpp` can be found together with Tutorial 4 in the web site!

3. Go to your `$EMBREE_SOURCES$/tutorials/common/scenegraph/` folder and rename the `materials.h` and `obj_loader.cpp` files (e.g., to `materials_original.h` and `obj_loader_original.cpp`).
4. Download the new files `materials.h` and `obj_loader.cpp` from the web site into folder `$EMBREE_SOURCES$/tutorials/common/scenegraph/`;

5. Build the `viewer` tutorial.

We will also use a new scene:

6. Download `veach_mis.zip` and extract it, making sure that the respective files (containing the `veach_mis` model, with extensions `.obj`, `.mtl` and `.ecs`) become available in the `$TUTORIALS_BUILD$/models` folder, where `$TUTORIALS_BUILD$` is the pathname of the folder where the `viewer` executable file is stored.
7. Verify your installation by opening a *shell* and from the `$TUTORIALS_BUILD$` folder executing

```
viewer –c models/veach_mis.ecs
```

The Scene Setup and Rendering Algorithm

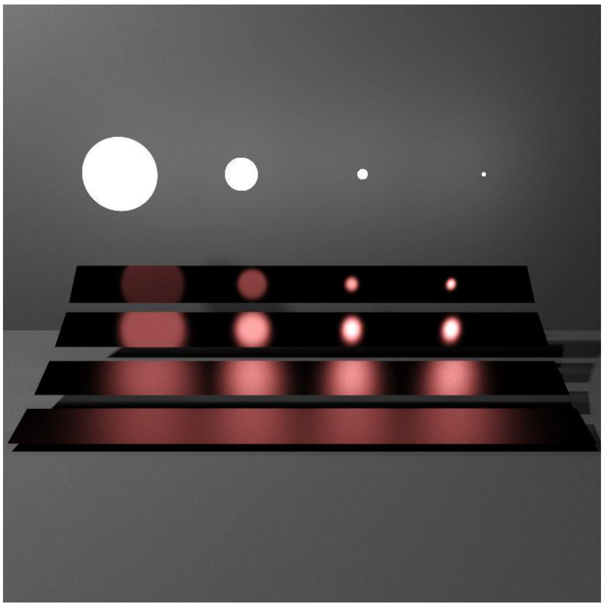

Figura 1 - Veach scene rendered using pbrt (not Embree)

The image you have rendered (Figure 1 includes a much higher quality rendering obtained with pbrt) is constituted by 4 plates with different glossinesses. The farthest away plate is purely specular (the BRDF is a Dirac delta: to any view direction corresponds a single incident direction), while the roughness of the surfaces increases for the other plates (the BRDF lobe for a given view direction becomes larger). You can also see that there are four different sizes light sources. This scene is directly inspired on the famous scene appearing on Eric Veach's Ph.D. dissertation, Stanford University, December 1997, entitled "Robust Monte Carlo Methods for Light Transport Simulation".

The rendering algorithm used is extremely simple: only paths of length 2 are shot. The primary ray is shot from the eye, through the image plane into the scene. Then a single secondary ray is shot (see figure 2). The question is: how to select the direction for this secondary ray?

Remember that reflected radiance at a point *p* is given by the integral over the hemisphere Ω of an integrand that is the product of three terms: the BRDF, the incident radiance and the cosine of the incident direction and the surface normal at *p* – see equation

$$L_r(p \rightarrow \omega_r) = \int_{\Omega} f_r(p, \omega_r \leftrightarrow \omega_i) L_i(p \leftarrow \omega_i) \cos(\vec{N}_x, \omega_i) d_{\omega_i}$$

Equação 1- Reflected radiance integral

One of the main issues when evaluating this integral using Monte Carlo techniques is which directions ω_i to select for sampling.

Importance sampling tells us that they should be selected according to how important they are for the value of the integral, i.e., we should select directions that maximize the value of the integrand and these directions must be selected according to a given probability distribution function (pdf).

The problem is that the integrand is the product of three terms and it is often impossible to build a pdf that follows the overall shape of the integrand (one of the main reasons is that in the general case the incident radiance L_i is unknown).

Consider Figure 2 below. If you are given the chance to STOCHASTICALLY select a single secondary ray, would you select it according to the light source pdf (solid angle represented in yellow) or to the BRDF lobe (solid angle in green)?

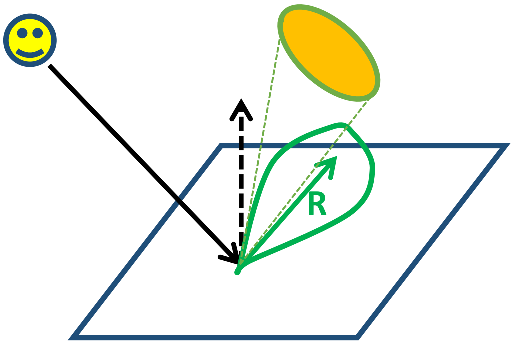

Figure 2- our algorithm setup. Should we select the sampling direction according to a pdf based on the lobe of the BRDF of according to the solid angle of the light source?

Sampling according to the Light Sources

Open the file `VI2_EmbreeT5_MIS.cpp` and verify that the selected sampling method is according to the light sources as seen below:

```
        // control the sample method
        typedef enum { LIGHT, BRDF, MIS } Sample_Method;

        Sample_Method method = LIGHT;
```

In function `secondaryRay()` you will see that the only secondary rays are shadow rays:

```
if (method == LIGHT) {
        Light_SampleRes ls;
        // do not divide color by lightPDF because it has already been factored in within
direct()
        L = direct(ray, stats, dg, material, sampler, Rdir, ls);
        color += L;
}
```

which are shot according to the light sources solid angles.

For which cases does this sampling method captures the light sources reflections on the plates? For the glossiest ones or for the more specular ones? For the ideally specular plate can you see any reflection? Why not?

Sampling according to the BRDF

In `VI2_EmbreeT5_MIS.cpp` make sure that the selected sampling method is according to the BRDF as seen below:

```
        // control the sample method
        typedef enum { LIGHT, BRDF, MIS } Sample_Method;

        Sample_Method method = BRDF;
```

In function `secondaryRay()` you will see that the secondary rays directions are now selected according to the BRDF solid angle:

```
else if (method == BRDF) {
  // for this scene materials are either diffuse or glossy; no further options
  if (reduce_max(material->Kd) > 0.f) {
        // get diffuse reflection sample: includes direction + probability
        const Vec2f sample = RandomSampler_get2D(sampler);
        const Sample3f KdSample = cosineSampleHemisphere(sample.x, sample.y, dg.Ns);

        /* initialize ray */
        sec_ray.org = Vec3fa(dg.P);
        sec_ray.dir = KdSample.v;
        ……

        /* intersect ray with scene */
        rtcIntersect(g_scene, sec_ray);
        L = Vec3fa(0.0f);
        // intersected the QuadLights added with add_4QuadLights
        for (l = 0; l < 4; l++) {
        if ((QuadLightID[l] != -1) && (sec_ray.geomID == QuadLightID[l])) {
                L = QuadLighRad[l];  break;
        }       }

        L *= float(pi) * material->Kd;
        color += L;
  }
  // specular reflection
  else if (reduce_max(material->Ks) > 0.f) {
        Sample3f KsSample;
```

```
    if (material->Ns >= 1.e4f) { // specular ideal
        KsSample.v = Rdir;
        KsSample.pdf = 1.f;
    } else {  // glossy
        const Vec2f sample = RandomSampler_get2D(sampler);
        KsSample = powerCosineSampleHemisphere(sample.x, sample.y, Rdir, material->Ns);
    }

    /* initialize ray */
    sec_ray.org = Vec3fa(dg.P);
    sec_ray.dir = KsSample.v;
    ……

    /* intersect ray with scene */
    rtcIntersect(g_scene, sec_ray);

    L = Vec3fa(0.0f);
    for (l = 0; l < 4; l++) {
    if ((QuadLightID[l] != -1) && (sec_ray.geomID == QuadLightID[l])) {
        L = QuadLighRad[l];  break;
    } }

    L *= material->Ks;
    if (material->Ns < 1.e4f) { // glossy lobe
        L *= powf(dot(sec_ray.dir, Rdir), material->Ns);
        L /= KsSample.pdf;
        L *= logf(material->Ns);
    }
    color += L;
  } // end specular reflection
} // end BRDF sampling
```

Can you see the light sources reflections on the specular plates now? Why?

What happens with the glossier plates and the smaller light sources? Why?

Multiple Importance Sampling

In `VI2_EmbreeT5_MIS.cpp` make sure that the selected sampling method is MIS as seen below:

```
    // control the sample method
    typedef enum { LIGHT, BRDF, MIS } Sample_Method;

    Sample_Method method = MIS;
```

In function `secondaryRay()` you will see that:

```
else if (method == MIS) {
    float weight;
    Vec3fa L;
    float which_dist = RandomSampler_get1D(sampler);
```

The random number `which_dist` will select whether light or BRDF sampling is used. But then the computed radiance is weighted by both probabilities of sampling.

If light sampling is selected then we have:

```
if (which_dist < 0.5f) {  // sample using the light sources
    Light_SampleRes ls;
    L = direct(ray, stats, dg, material, sampler, Rdir, ls);
```

```
        // compute the probability wit which this direction was found
        lightPDF = ls.pdf;

        // get the probability of this direction if it had been sampled from the BRDF
        if (reduce_max(material->Kd) > 0.f) { // if it is diffuse
                brdfPDF = cosineSampleHemispherePDF(ls.dir);
        }
        else {  // if it is specular
          if (material->Ns >= 1.e4f) { // specular ideal
                brdfPDF = 0.f;
          } else {  // glossy
                brdfPDF = powerCosineSampleHemispherePDF(ls.dir, material->Ns);
          }
        }
        // compute the MIS weight
        // weight = ...
}
```

Note that the weight calculation is not completed. It is up to you to complete that line. Remember: the weight is equal to the ratio between the PDF that was actually used (light in this case) and the sum of all possible probability densities (light plus BRDF in this case).

If BDRF sampling is selected then we have:

```
else { // sample the BRDF
        if (reduce_max(material->Kd) > 0.f) { // diffuse
                const Vec2f sample = RandomSampler_get2D(sampler);
                const Sample3f KdSample = cosineSampleHemisphere(sample.x, sample.y, dg.Ns);
                brdfPDF = KdSample.pdf;

                /* from here is the same as above … */
                ……
        }
        else if (reduce_max(material->Ks) > 0.f) { // specular reflection
          Sample3f KsSample;
          if (material->Ns >= 1.e4f) { // specular ideal
                KsSample.v = Rdir;
                KsSample.pdf = 1.f;
          } else {  // glossy
                const Vec2f sample = RandomSampler_get2D(sampler);
                KsSample = powerCosineSampleHemisphere(sample.x, sample.y, Rdir, material->Ns);
          }
          brdfPDF = KsSample.pdf ;

          /* from here is the same as above … */
          ……
        } // end specular reflection

        // what would be the probability density of sampling this same direction according to
        // the light sources?
        lightPDF = directPDF(sec_ray.dir, dg, sampler);
        // compute the MIS weight
        // weight = ...
}
```

Note that the weight calculation is once agein incomplete. You do it using the same rational as above, but remembering that now the BRDF was used for sampling.

Finally the computed radiance `L` is weighted by the computed weight:

```
        // weight radiance L by the sum of all sampling probabilities
        L *= weight;
```

and then L is divided by the probability with which this particular sampling (light or BRDF) was selected:

```
	// divide the radiance by the probability with which the particular sampling PDF was selected
	//L /= ...;
```

That division is not completed. You complete it. With which probability did we select either light or BRDF sampling?

Build the viewer and run it. What do you think of the results? Are all highlights correct now?

You can press ‘l’ in runtime to change between sampling methods: `LIGHT` -> `BRDF` -> `MIS` -> …