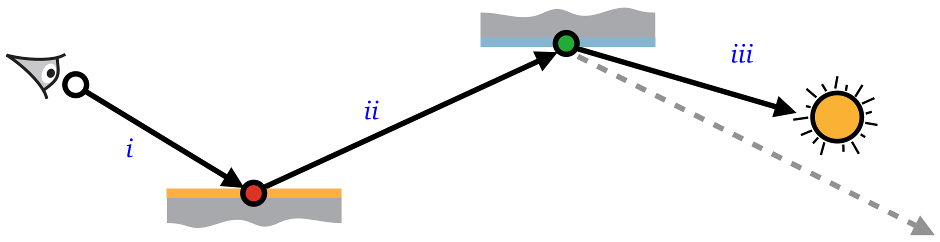
0.1 Implicit Path Tracing with BRDF importance sampling

Overview The implicit path tracing (IPT) is an algorithm that can simulate global illumination in virtual scenes.It works by reverse tracing the rays from observer's eye or camera and allowing it to undergo several reflection(regular and irregular) until it hits the light source, in order to render the radiance of light at the point of gaze.There exists rays that might undergo several bounces before it hits the light source or even miss it (e.g., gray dotted arrows in the overview figure), such rays generally do not contribute significantly to the radiance. Hence the implementation allows ray-bounces up to depth of 2 in order to render image with bounded space and time complexity.



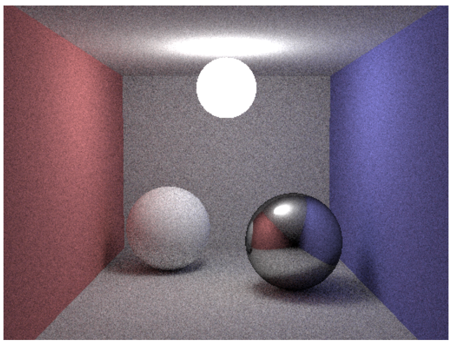
Implicit path tracing traces paths that start from the eye (*i*), scatter at surfaces in the scene (*ii*), in the blind hope that they eventually (*iii*), hit a light source.

To render objects of different texture namely diffused or glossy with its glossiness ranging from near perfect specular reflection to almost diffused glossy, sampling at the point of intersection has to be done according to the Bi-directional reflectance distribution function(BRDF) .ie BRDF importance sampling.

Scene Setup The scene rendered includes several objects which are either sphere or a part of very large spheres.No object except the source emits light.The source is a sphere with color vector RGB(1.0,1.0,1.0) and a radiance of RGB(10,10,10).The walls are made up of outer surface of very large diffused spheres.There exist no wall on the front side ie. the scene is not closed.Few light rays that have direction perpendicular to the wall on the back side, will escape the scene except the one that is in line with the camera.Also there are several other rays that might hit one of the 4 bounding surface behind the camera, and will not contribute to render the scene.

Apart from the walls, the scene also consists of 2 relatively small spheres, of which one is diffused and the other is glossy.The glossiness of the sphere is defined by its phong factor.

The whole scene is placed in vacuum and phenomena such as refraction and scattering can be ignored.



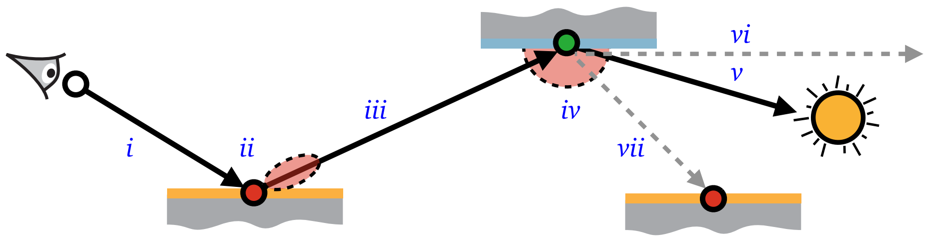
A sample image depicting the scene that will be rendered.

Algorithm Details A pixel in the image depicts a small portion of the real world.The intensity of that pixel is basically the outgoing radiance at the point of gaze in real world that represents that small portion of real world.The radiance is determined by two separate phenomena which are reflection and emission of light.While emission may be due to either black body radiation or luminescence,radiance due to reflection depends on the light coming from the surrounding region and the texture of the object itself.

In real world, light rays begin at the source carrying radiance and hit objects to illuminate them.But this process of illumination is not just the direct interaction with the source and object.The light ray that illuminates an object might bounce of several surfaces and each bounce imparts some changes in the radiance of light being carried ahead.This phenomenon is responsible for effects like color bleed.

While rendering, it is computationally very expensive to mimic the exact same process that happens in nature ie. starting at source and considering only those light rays that hit the camera, as many of the light rays will not hit the camera resulting in wastage of computational resource.

In the implementation, the reverse process is carried out, which involves throwing a ray from the camera and then tracing back to light source within the permitted number of bounces.The rays here are the exact reverse of the light rays that would have traced the same path.These rays can be termed as "query rays" as they hit a point and return its radiance.



Implicit path tracing with BRDF importance sampling handles tracing of paths that start from the eye (*i*), scattering at a glossy surfaces in the scene (*ii*), reflected ray (*iii*), scattering at a diffused surface (*iv*), hitting a light source(*v*), missing a light source(*vi*), exceeding two bounces(*vii*).

So to determine the radiance of a point we throw a query ray towards that point from the camera.Now since the radiance at that point is determined by intensities at several other points, a single throw of query ray triggers throwing of another query ray starting from the point of gaze to a different point in the environment.The only time when this recursive querying does not happen is when the query ray hits the light source,instead it returns the radiance of the light source.

The process of determining the direction in which the second or subsequent query rays is to be thrown, is called sampling.To account for different textures of objects sampling has to be done according to different sampling techniques relevant to the object texture. (As depicted in the image by the pink lobes).

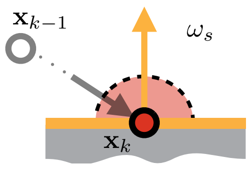
Texture is a term that depicts where and how much does the object reflect the incident light rays and such behavior is represented by Bi-direction reflectance distribution function(BRDF).So when sampling is done keeping in mind the texture of the object, it is called BRDF importance sampling.

$$L(\mathbf{x},\omega_o) = L_e(\mathbf{x},\omega_o) + \int_{\Omega_{\vec{\mathbf{n}}}} L(\mathbf{x}',-\omega_i)\, f_r(\mathbf{x},\omega_i,\omega_o)\, (\vec{\mathbf{n}}\cdot\omega_i)\, \mathrm{d}\omega_i\,, \tag{1}$$

The concept of IPT with BRDF importance sampling can be represented by eqn(1).On the

LHS, $L(\mathbf{x}, \omega_o)$ represents outgoing radiance at a given point $\mathbf{x}$ in the scene, along a direction represented by solid angle ω_o. On the RHS, L_e represents radiance due to light emission, which in case of our scene, turns out to be zero for all objects except the source. The second part is an integration over all direction in hemisphere above point $\mathbf{x}$, of all the incoming radiance projected onto normal at point x, times fraction of incoming radiance that is reflected in outgoing direction. $f_r(\mathbf{x}, \omega_i, \omega_o)$ is called the BRDF which depicts the fraction of incoming radiance coming from ω_i reflected in ω_o direction.

Object-Light Interaction The amount off radiance that is reflected off a surface is determined by the BRDF as discussed above.



Scattering at diffused surface.

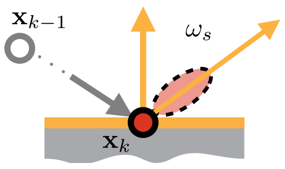
Diffused object reflect incoming light in all directions in the hemisphere about point $\mathbf{x}$, as shown in the BRDF of diffused surface image for diffused objects image.The fraction of incoming radiance reflect in each direction is

$$f_r(\mathbf{x}, \omega_i, \omega_o) = (1/\pi) \tag{2}$$

When sampling is done on a diffused object , it is uniform hemispherical importance sampling for which the probability distribution function

$$pdf = (1/2\pi). \tag{3}$$

Glossy objects reflect incoming radiance in a cosine power lobe which has perfectly reflected ray as its axis as shown in the BRDF of glossy surface.The thickness of the cosine lobe is determined by phong factor n. The BRDF for glossy objects is



Scattering at glossy surface.

$$f_r(\mathbf{x}, \omega_i, \omega_o) = (n+1)cos^n(\theta_r)/(2\pi) \tag{4}$$

where θ_r is the angle between sampled ray and cosine lobe axis. The pdf is inverse of the BRDF in this case.

$$pdf = (2\pi)/((n+1)cos^n(\theta_r)) \tag{5}$$

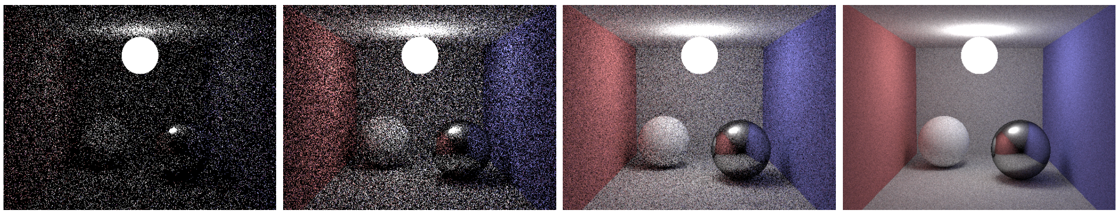
Implementation Details Technically, the IPT algorithm is a Monte Carlo (MC) estimator of the rendering equation [Kajiya86] The implementation determines the approximate value of the outgoing radiance at the point of gaze by computing integral in eqn(1) using monte carlo estimator. eqn(6) below represents the 1 sample estimate of eqn(1).

$$\mathrm{E}_{\Omega}^{1\mathrm{spp}}\left[L(\mathbf{x}, \omega_o)\right] = L_e(\mathbf{x}, \omega_o) + L(\mathbf{x}', -\omega_s)\, f_r(\mathbf{x}, \omega_s, \omega_o)\, (\vec{\mathbf{n}} \cdot \omega_s)/pdf(\omega_s) \tag{6}$$

$\mathrm{E}_{\Omega}^{1\mathrm{spp}}$ is equivalent to throwing a query ray and computing the intensity of pixel along a single path where the direction at each bounce is determined through BRDF importance sampling with relevant probability distribution function.

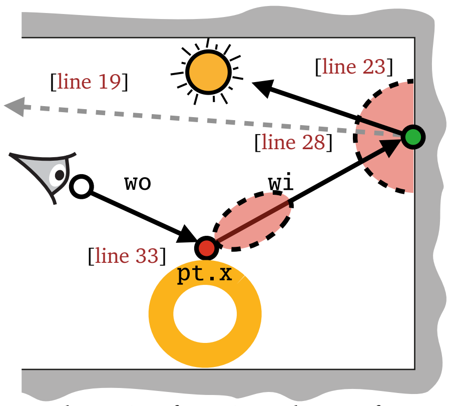
By the end of a single query ray throw at the point of gaze we get the radiance at point of gaze due to a single path taken by the query ray.However the actual radiance of point of gaze is determined by several paths that can be taken in the environment.Thus the single query ray throw process is performed several times over a point of gaze and then an average of all the intensities is taken to render the actual radiance.

The convergence images below depicts that the image has a high variance when computed over less samples but eventually converges to an image with low variance as the samples taken are increased.

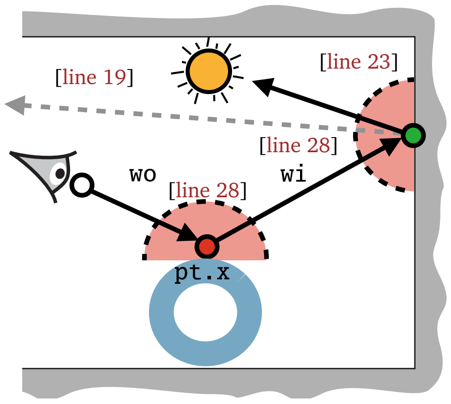


Convergence : As we increase the MC sample rate, variance decreases and IPT with BRDF importance sampling convergences to the correct solution.

```
 1 rgb Lo(const ray &wo,const int bounce){
 2  // variable to store intersection data
 3  surface pt;
 4
 5  //variable to store pdf
 6  double pdf;
 7
 8  //if the number of bounces left is zero,then return
 9  if (bounce==0){
10  return rgb(0.0,0.0,0.0);
11  }
12
13  else{
14   // trace the ray wo into the scene
15   bool hit = intersect(wo, &pt);
16
17   // if ray exits the scene
18   if (!hit)
19   return rgb(0.0,0.0,0.0);
20
21   // hit a light, return emission...
22   if( pt.e != rgb(0.0, 0.0, 0.0) )
23   return pt.color;
24
25   //perform sampling depending on the
26   //type of surface and set pdf
27   if(pt.isdiffused()){
28   vec3 wi = sampleHemisphere(pt.n);
29   pdf=setdiffusepdf();
30   }
31
32   else{
33   vec3 wi = samplePhongLobe(pt.n,wo,pt.phong);
34   pdf=setglossypdf(pt.n,wo,pt.phong);
35   }
36
37   // compute recursive integral
38   return  pt.color * pt.fr(wi,wo) * Lo(ray(pt.x,
       wi),bounce-1)*(1/pdf);
39  }
40 }
```



When point of gaze is on glossy surface



When point of gaze is on diffused surface

The pseudo code above assumes that the implementation of a basic ray tracer is done which includes function like intersect that can intersect rays and return surface intersection data such as intersection point,type of surface,surface's phong factor, normal and a function fr to return the BRDF at x.

Implicit Path Tracing Reference

[Kajiya86] James T. Kajiya. *The Rendering Equation*. Proceedings of the 13th annual conference on Computer graphics and interactive techniques (SIGGRAPH). 1986