# Subsurface Rendering Using Monte Carlo Methods

XIANGMING KA

This paper demonstrates the improvements of my pervious ray tracer that includes path tracing, subsurface rendering, and importance sampling using Monte Carlo methods. The path tracing based on Kajiya's rendering equation enables global illumination and supports further extensions. In the rest of this paper, I focus on subsurface rendering to present subsurface scattering, emission, and absorption effects. And I implemented importance sampling volume rendering methods based on Wojciech Jarosz's papers that used several Monte Carlo methods to simulate light transport in scenes with participating media.

Additional Key Words and Phrases: Monte Carlo methods, volume rendering, path tracing

**ACM Reference Format:**
Xiangming Ka. 2020. Subsurface Rendering Using Monte Carlo Methods. 1, 1 (January 2020), 5 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

## 1 PATH TRACING

I extended my previous ray tracer to support path tracing based on rendering equation. Every time the ray hit a non-specular object, the path tracer will trace a ray in a uniformed random direction in the hemisphere.

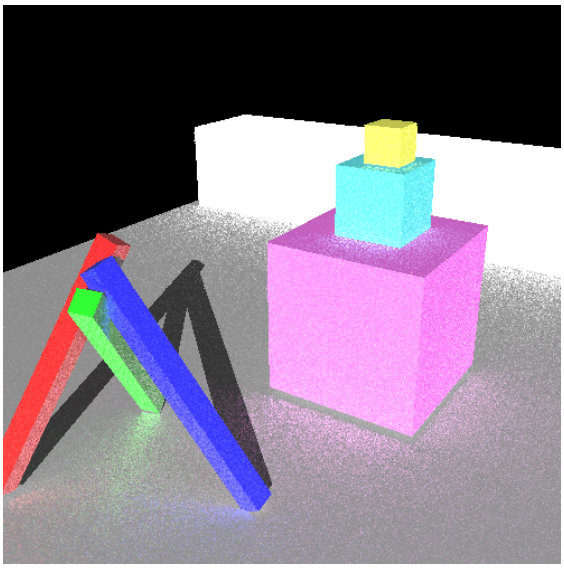


Fig. 1. Scene rendered using path tracing by 10 samples each bounce

Author's address: Xiangming Ka.


XXXX-XXXX/2020/1-ART $15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

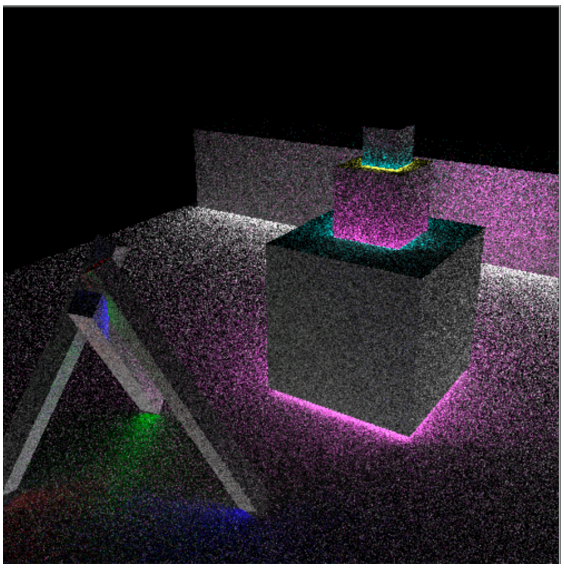


Fig. 2. Rendered only environmental light

There are still a lot of space for improvement, but it is good enough to support the focus of this paper–volume rendering.

## 2 BASIC VOLUME RENDERING

In this section, I am going to implement a transitional way to present volume rendering. This method is functional working but inefficient and biased. In the next section I will improve this method to get better efficient.

The main properties of a participating media are defined by the following variables:

- Absorption : $-\mu_a(x)L(x,\omega)$
- Emission : $\mu_a(x)L_e(x,\omega)$
- Out-scattering : $-\mu_s(x)L(x,\omega)$
- In-scattering : $\mu_s(x)L_s(x,\omega)$

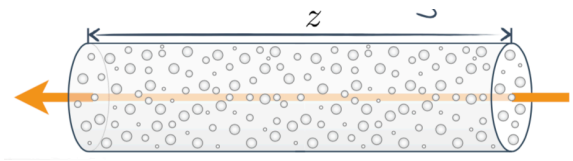
Assume the sum radiance of light that transfers from y to x is $L(x,\omega)$, and the radiance can be calculated by the following formulas:

$$\frac{\mathrm{d}L(x,\omega)}{\mathrm{d}z} = -\mu_a(x)L(x,\omega) - \mu_s(x)L(x,\omega) + \mu_a(x)L_e(x,\omega) + \mu_s(x)L_s(x,\omega)) \tag{1}$$

$$L(x,\omega) = \int_0^z T(x,y)[\mu_a(y)L_e(y,\omega) + \mu_s(y)L_s(y,\omega)]dy \tag{2}$$

$$T(x,y) = e^{-\int_0^y \mu_t(s)ds} \tag{3}$$

T(x, y) is the fraction of light that makes it from y to x.




## 2.1 Absorption effect

The absorption part of the classical radiative transport theory is defined as:

$$L(x,\omega) = (x-z)\mu_a L(z,\omega) \tag{4}$$

It is based on the assumption that the medium has uniformed density, so we only need to calculate the length that light transfers in the volume. The result is as following.

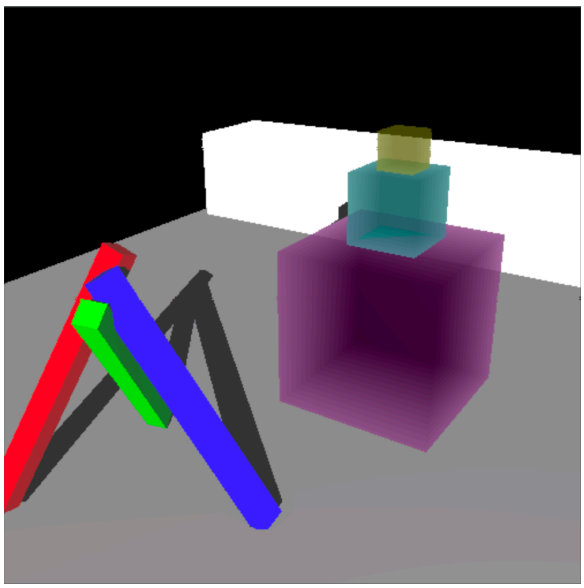


Fig. 3. Absorption effect in homogeneous volume

Then I extended it to integrate heterogenous medium by integrating the remaining light along the path as the formula demonstrated:

$$L(x,\omega) = \int_0^z (\int_0^y \mu_a(s)\ \mathrm{d}s) L(y,\omega)\ \mathrm{d}y \tag{5}$$

## 2.2 Emission effect

The emission part's contribution of the radiative transport is defined as following:

$$L(x,\omega) = \int_0^z \mu_a(y) L_e(y,\omega)\ \mathrm{d}y \tag{6}$$

When take into account both absorption and emission, I accumulated the contribution by a certain pace along the path, and the integration should be

$$L(x,\omega) = \int_0^z (\int_0^y \mu_a(s)\ \mathrm{d}s)[\mu_a(y)L_e(y,\omega)]dy + (x-z)\mu_a L(z,\omega) \tag{7}$$

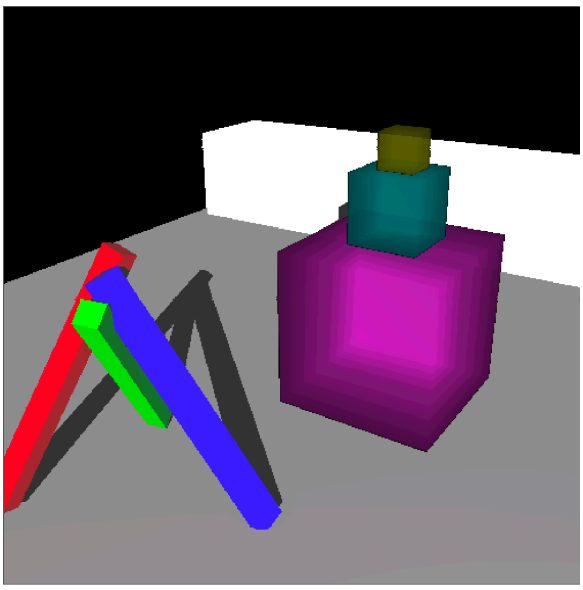


Fig. 4. Emission effect alone

**ALGORITHM 1:** find_attenuation

**Input:** $\tau_0$,$\tau_1$,$s_0$,h
**Output:** $s$
$T = ln(\tau_0)$;
$T_1 = ln(\tau_1)$;
**for** $s = s_0; T < T_1; s+ = h$ **do**
  $f = \sigma_t(s)$;
  $T+ = hf$ ;
  $s- = (T - T_1)/f$
**end**
return s ;

**ALGORITHM 2:** radiance_estimator

**Input:** x,$\omega$,$\xi$-list
**Output:** $L$
$s_{max}$, $L_0$ = trace_ray();
S = 0;
n = $\xi - list.length$ ;
**for** $i = 0; i < n; i++$ **do**
  $\xi$ = $\xi$-list[i];
  s = find_attenuation($\omega$, $s$, $s_{max}$, $h$);
  **if** $s$<$s_{max}$ **then**
    S += $\epsilon(x - s\omega)/\sigma_t(x - s\omega)$;
  **else**
    break;
  **end**
**end**
return $(S + (n - i)L_0)/n$ ;

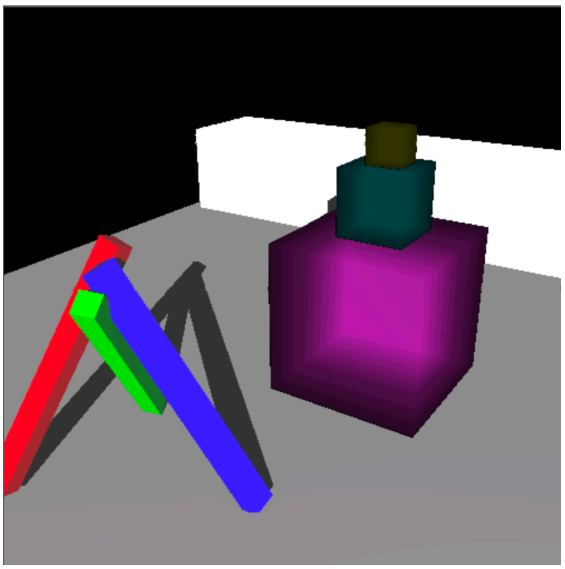


Fig. 5. Both absorption and emission effects

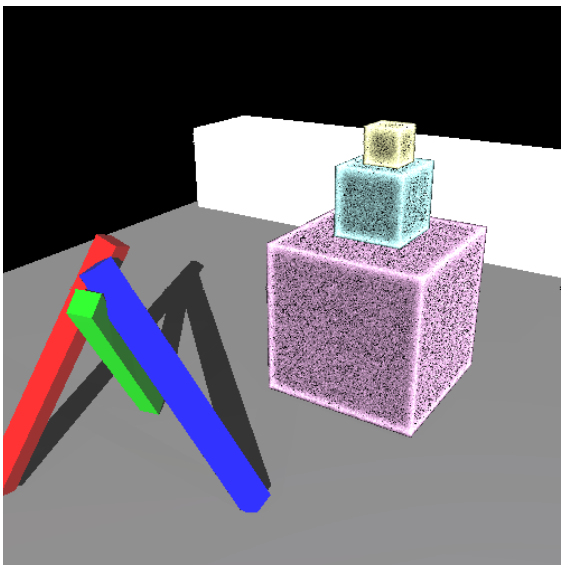


Fig. 6. In-scattering effect by 10 samples each pace

**ALGORITHM 3:** ray_scattering

**Input:** x,$\omega$,$\xi$
**Output:** $L$
$L_0$ = trace_ray();
S = 0;
y = x;
**for** $i = 0; i\xi < z; i++$ **do**
  y += $\xi$;
  **for** $j = 0; j < number\_of\_samples; j++$ **do**
    $\omega'$ = sample_phase_function() ;
    $L_s$ = radiance_estimator($x - s\omega$, $\omega'$, $new\_\xi\_list()$ );
    $s+ = (\epsilon(x - s\omega)/\sigma_t(x - s\omega))$;
  **end**
  S += $\epsilon(x - s\omega)/\sigma_t(x - s\omega)$;
**end**
return $S/i * z$ ;

### 2.3 Out-scattering and in-scattering effects

The out-scattering effect is just like absorption, but the in-scattering is trickier. The main idea is that as I accumulating the contribution along the path by a fixed pace, I sampled the light at uniformed random directions each pace and integrated those samples as (9). The integration is demonstrated as the follow formula:

$$L(x, \omega) = \int_0^z \mu_s(y) L_s(y, \omega)]dy \tag{8}$$

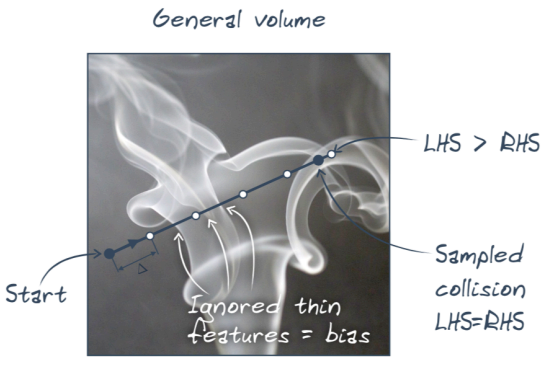
$$L_s(y, \omega) = \int_{S^2} f_p(\omega, \bar{\omega}) L(y, \bar{\omega}) d\bar{\omega} \tag{9}$$

I first implemented the ray matching algorithm to calculate the integration:

## 3 ADVANCED VOLUME RENDERING

Using the naive method also known as ray marching, the image is extremely noisy. It is because this approach sampled every scattering event by a fixed distance along the ray, but only a small part of those samples contributed to the final result. And ray marching is a biased method because it lost a lot of details while sampling in heterogeneous volumes.

$$\int_0^t \mu_t(s)ds \neq \sum_{i=1}^{k} \mu_{t,i}\Delta \tag{10}$$




In this section I am going to implement importance sampling methods to improve the quality and efficiency. Importance-sampling methods' main idea is to add weigh to every sample based on BRDF along the path.

**ALGORITHM 4:** sample_distance

**Input:** x,$\omega$,$\sigma_{max}$
**Output:** $s$
s = 0;
**while** *true* **do**
  s += $-ln(1 - rand())/\sigma_{max}$;
  **if** *rand()* < $\sigma_t(x - s\omega)/\sigma_{max}$ **then**
    break ;
  **end**
**end**
return s ;

### 3.1 Woodcock tracking

The idea is to take many steps as if the medium were denser than it is, and then randomly choose to return that distance or take another step.

where $\sigma_{max}$ is an upper bound for $\sigma_t$ over the whole ray. It stops in denser areas more often, since the random walk terminates with probability proportional to $\sigma_t(x)$, and it will never stop where $\sigma_t(x) = 0$. And it seems plausible that the probability decreases with distance since more or larger steps are required to go farther. But it can indeed be proven to produce values of $s$ with probability density.

$$p(s) = \sigma_t(x)exp(-\int_0^s \sigma_t(s')ds') \tag{11}$$

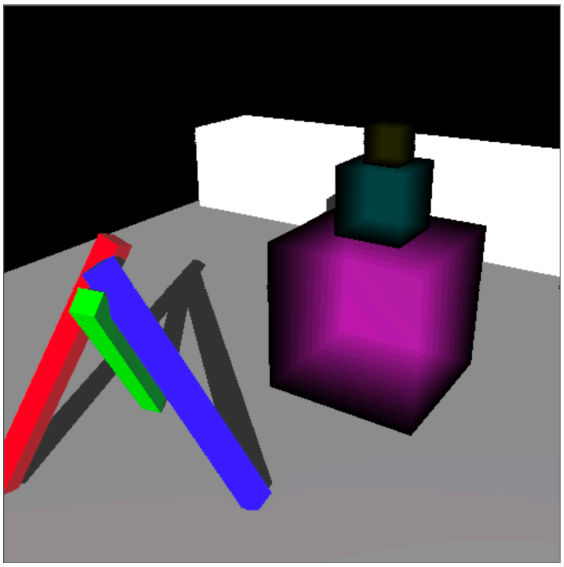


Fig. 7. Both absorption and emission effects using Woodcock tracking

The compression shows that the result using Woodcock tracking is smoother than Ray marching method.

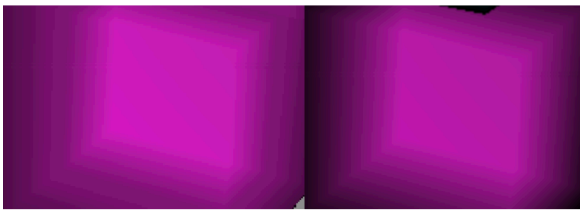


Fig. 8. Compression of Ray marching and Woodcock tracking

**ALGORITHM 5:** importance_sampling_integral

**Input:** x,$\omega$,$\xi$
**Output:** $L$
$L_0$ = trace_ray();
S = 0;
y = x;
**for** $i = 0; i\xi < z; i++$ **do**
  y += $\xi$;
  **for** $j = 0; j < number\_of\_samples; j++$ **do**
    $\omega'$= sample_phase_function() ;
    $L_s$= radiance_estimator($x - s\omega$, $\omega'$, $new\_\xi\_list()$ );
    s+ = $(\epsilon(x - s\omega + \sigma_s L_s)/\sigma_t(x - s\omega))$;
  **end**
  S += $\epsilon(x - s\omega)/\sigma_t(x - s\omega)$;
**end**
return $S/i * z$ ;

### 3.2 Advanced Monte Carlo estimator for the scattering integral

We now know how to handle absorption and emission in the general case. The remaining issue is how to compute scattering. As we've observed before, out-scattering behaves exactly like absorption, as far as attenuation is concerned. Similarly, in-scattering behaves exactly like emission, as far as the integral along the ray is concerned. So we can use the method from the previous section but modify the estimator to importance sample.

$$\epsilon_s(x', \omega) = \sigma_s(x') \int_{4\pi} f_p(x', \omega, \omega')L(x', \omega')d\omega' \tag{12}$$

$$g(\omega') = \frac{\sigma_s(x')f_p(x', \omega, \omega')L(x', \omega')}{p(\omega')} = \sigma_s(x')L(x', \omega') \tag{13}$$

The first thing to try is importance sampling by the phase function(This means that whenever we propose a particular phase function, we also need to come up with a procedure for sampling it.); so in this case $p(\omega') = fp(x', \omega, \omega')$. Then the estimator is

As the compression shows, the image quality improved dramatically using this method. It may have better result if I have a more complicated scene.

Using this method, we can avoid wasting too many samples on the positions where less contribute to the whole radiance thus improve efficiency. There are many other methods to importance sample in this situation, and I think using machine learning may be a promising method.

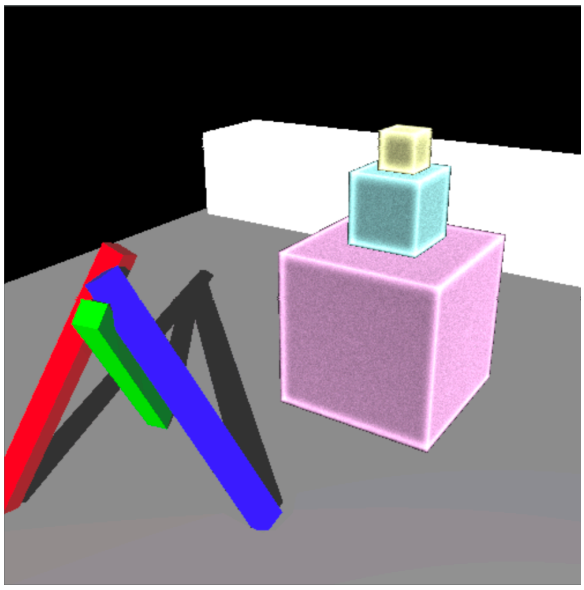


Fig. 9. Light scattering using importance sampling

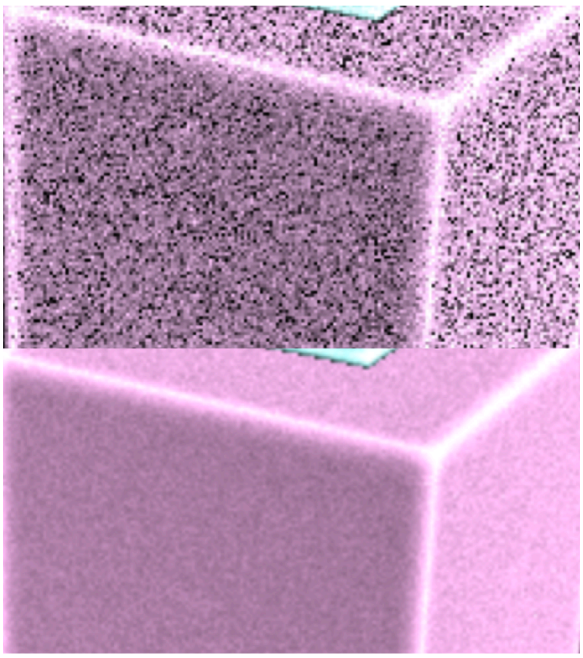


Fig. 10. Compression of even sampling and importance sampling

To sum up, in this assignment I first extended my previous ray tracer to path tracer. Then I implemented four kinds of volume or subsurface rendering effects to make it more realistic. Finally I improved the estimators by using importance sampling methods.

There are still a lot of space to research, but I did my best to show you what I could do. Thanks for reading, hopefully you like it.