7

The Photon Mapping Method

"I get by with a little help from my friends."

—John Lennon, 1940–1980

PHOTON mapping is a practical approach for computing global illumination within complex environments. Much like irradiance caching methods, photon mapping caches and reuses illumination information in the scene for efficiency. Photon mapping has also been successfully applied for computing lighting within, and in the presence of, participating media. In this chapter we briefly introduce the photon mapping technique. This sets the foundation for our contributions in the next chapter, which make volumetric photon mapping practical.

7.1 Algorithm Overview

Photon mapping, introduced by Jensen [1995; 1996; 1997; 1998; 2001], is a practical approach for computing global illumination. At a high level, the algorithm consists of two main steps:

Algorithm 7.1: PHOTONMAPPING()

```
PhotonTracing();
RenderUsingPhotonMap();
```

In the first step, a lighting simulation is performed by tracing packets of energy, or *photons*, from light sources and storing these photons as they scatter within the scene. This processes

Algorithm 7.2: PHOTONTRACING()

$n_e = 0$;
repeat
    (l, $pdf(l)$) = CHOOSELIGHT();
    $(\mathbf{x}_p, \vec{\omega}_p, \Phi_p)$ = GENERATEPHOTON(l);
    TRACEPHOTON($\mathbf{x}_p, \vec{\omega}_p, \frac{\Phi_p}{pdf(l)}$);
    n_e += 1;
until *photon map full* ;
Scale power of all photons by $\frac{1}{n_e}$;

results in a set of *photon maps*, which can be used to efficiently query lighting information. In the second pass, the final image is rendered using Monte Carlo ray tracing. This rendering step is made more efficient by exploiting the lighting information cached in the photon map. Radiance can be evaluated at arbitrary points in the scene by locally computing the photon density within the photon map.

7.2 Photon Tracing

In the first pass, photons are emitted from light sources and traced through the scene just as rays are in ray tracing. The photon tracing pass is summarized in Algorithm 7.2. We describe this procedure in the following sections.

7.2.1 Photon Emission

Intuitively, photon mapping works by splitting the energy emitted by each light source into discrete packets called photons[1]. A number of photons are emitted from a light source, and each photon represents a fraction of the total power of the light. Photon tracing starts by generating a random photon at a light source. Each photon p includes a position and direction $\mathbf{x}_p, \vec{\omega}_p$, as well as an associated power, Φ_p. Any type of light source can be used to emit photons by providing an appropriate GENERATEPHOTON procedure. Jensen provides details on how to generate photons for emission from a variety of commonly used light sources [2001]. If multiple

[1] A “photon” in the photon mapping technique actually corresponds to a collection of physical photons.

light sources are present, each time a photon is emitted, Russian roulette is used to choose the emitting light source. More photons can be shot from brighter light sources by basing the roulette probabilities on the power of the lights.

7.2.2 Photon Scattering

After a photon is generated for emission at a light source it is traced through the scene. This process is encompassed in the TRACEPHOTON function and proceeds analogously to tracing rays. When a photon intersects a surface it is either scattered or absorbed. Russian roulette is used to probabilistically choose which of these events occurs, and the surface's material properties determine the corresponding probabilities. The TRACEPHOTON function terminates once the photon is absorbed or it propagates out of the scene. If reflection or transmission occur, the photon is scattered according to the BRDF. For perfect mirror surfaces, the scattered direction is the mirror reflection direction, whereas for perfect Lambertian surfaces the direction is computed according to a cosine distribution. It is possible to importance sample a scattering direction from a wide range of other BRDFs as well. If importance sampling the BRDF is not possible then a random scattering direction can be chosen, and the photon's power then needs to be scaled according to the BRDF.

7.2.3 Photon Storage

Whenever a photon hits a non-specular surface, it is stored in a global data structure called a photon map. Photons hitting perfectly specular surfaces are not stored since this form of illumination can be more effectively computed using Monte Carlo ray tracing during the rendering pass. During rendering, illumination information is queried by performing range searches within the photon map. The photons are inserted into a balanced kd-tree acceleration structure in order to perform these range searches efficiently.

7.2.4 Importance-Driven Photon Mapping

The photon tracing procedure described so far is view-independent. This means that, if the camera is aimed at a relatively small portion of the scene, many photons may be wasted in "unimportant" regions of the scene. Peter and Pietrek [1998] used the concept of importance to concentrate photons in the parts of the scene where they contribute most to the final image. Before tracing photons, they emitted importance particles (or "importons") from the camera and stored these in an importance map as they scattered at non-specular surfaces. The importance map can be used to guide the reflection of photons [Peter and Pietrek, 1998], or to determine the probability of storing each photon [Suykens and Willems, 2000; Keller and Wald, 2000]. These approaches can dramatically reduce the required number of stored photons; however, the number of emitted photons remains high. Peter and Pietrek [1998] and Christensen [Jensen et al., 2001a] suggested ways to address this by using a set of "test" photons to determine important emission directions; however, the robustness of these techniques has not been well tested in practice. Christensen [2003] provides an in-depth survey of importance within computer graphics.

7.3 Radiance Estimation

After photon tracing finishes, the collection of stored photons represents the incident illumination (flux) on surfaces. Each photon represents a fraction of the flux emitted either directly by a light source or indirectly through intermediate scattering events.

Reflected radiance at a surface is computed used Equation 2.14:

$$L(\mathbf{x}\rightarrow\vec{\omega}) = \int_{\Omega} f_r(\mathbf{x},\vec{\omega}'\leftrightarrow\vec{\omega})\, L(\mathbf{x}\leftarrow\vec{\omega}')\,(\vec{\mathbf{n}}\cdot\vec{\omega}')\; d\vec{\omega}'. \tag{7.1}$$

We can relate this to the flux stored in the photon map using Equation 2.2, which expresses the relationship between radiance and flux:

$$L(\mathbf{x},\vec{\omega}) = \frac{d^2\Phi(\mathbf{x},\vec{\omega})}{(\vec{\mathbf{n}}\cdot\vec{\omega})\; d\vec{\omega}\; dA(\mathbf{x})}. \tag{7.2}$$

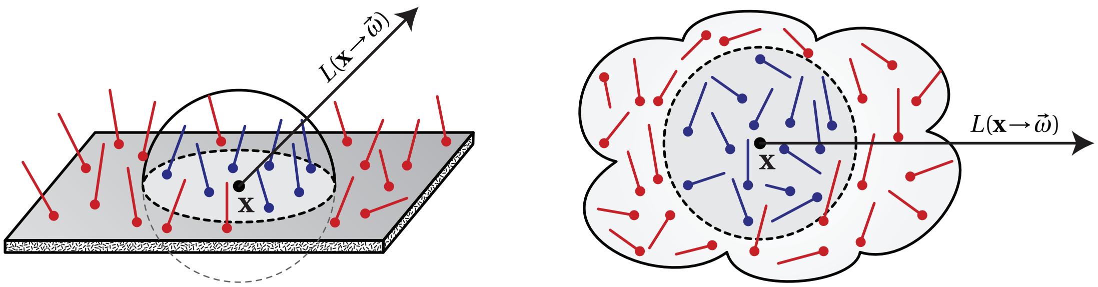

Figure 7.1: The radiance estimate (Equation 7.5) and the volumetric radiance estimate (Equation 7.10) compute outgoing radiance using density estimation by finding the nearest k photons to the query location $\mathbf{x}$. On surfaces, the contribution from each photon is weighted by the BRDF, accumulated, and divided by the *projected surface area* of the bounding sphere. Within participating media, each photon is instead weighted by the phase function, accumulated, and the result is divided by the *volume* of the bounding sphere.

By combining these expressions we arrive at:

$$L(\mathbf{x}\to\vec{\omega}) = \int_\Omega f_r(\mathbf{x},\vec{\omega}'\leftrightarrow\vec{\omega})\,\frac{d^2\Phi(\mathbf{x},\vec{\omega}')}{(\vec{\mathbf{n}}\cdot\vec{\omega}')\,d\vec{\omega}'\,d\mathcal{A}(\mathbf{x})}\,(\vec{\mathbf{n}}\cdot\vec{\omega}')\,d\vec{\omega}', \tag{7.3}$$

$$= \int_\Omega f_r(\mathbf{x},\vec{\omega}'\leftrightarrow\vec{\omega})\,\frac{d^2\Phi(\mathbf{x},\vec{\omega}')}{d\vec{\omega}'\,d\mathcal{A}(\mathbf{x})}\,d\vec{\omega}'. \tag{7.4}$$

This integral can be approximated using the photon map. Since each photon p carries flux Φ_p, the local density of the photons can be used to estimate the above integral. The density is computed by adding up the flux of the nearest photons in a small local region and dividing by the projected area $\mathcal{A}$ of the sphere containing these photons (see Figure 7.1). The reflected radiance at a point $\mathbf{x}$ in direction $\vec{\omega}$ can be estimated using the *radiance estimate* as

$$L(\mathbf{x}\to\vec{\omega}) \approx \sum_{p=1}^{k} f_r(\mathbf{x},-\vec{\omega}_p\leftrightarrow\vec{\omega})\,\frac{\Phi_p(\mathbf{x}\leftarrow-\vec{\omega}_p)}{\mathcal{A}(\mathbf{x})}, \tag{7.5}$$

where $\mathcal{A}(\mathbf{x}) = \pi r(\mathbf{x})^2$ is a surface area of a small region around $\mathbf{x}$.

This density estimation can be computed in a number of ways. The simplest approach sets r to a constant value over the whole scene. This approach would involve finding all photons p within a constant radius r sphere from the shading location $\mathbf{x}$ and applying the above expression. Since photons lie on surfaces, the surface area $\mathcal{A}(\mathbf{x})$ corresponds to projecting the sphere onto the surface and using the area of the resulting circle, $\mathcal{A}(\mathbf{x}) = \pi r^2$. The drawback of this simple approach is that a single radius may not be well suited for all regions of the scene. Jensen instead proposed using the nearest neighbor method to estimate the local density. The nearest neighbor method finds the nearest k photons from the query location $\mathbf{x}$ and sets the radius equal to the distance to the k^{th} photon from $\mathbf{x}$, $r(\mathbf{x}) = d_k(\mathbf{x})$. With this approach the search radius is allowed to adapt to the local density of photons.

The radiance estimate in Equation 7.5 can be further improved in a number of ways. A typical improvement is to weight the contribution of photons differently within the search radius. Weighting the photon contributions with a smooth filtering kernel smooths the results while reducing overblurring. This technique is called the generalized nearest neighbor method within the density estimation literature [Silverman, 1986]. We review density estimation in more detail in Appendix C.

7.4 Participating Media

In reality, light interacts not only with surfaces but also scatters and gets absorbed by the medium between surfaces. Jensen and Christensen extended the photon mapping method to simulate global illumination within participating media [1998]. This extension requires modification to both the photon tracing and the rendering portion of the photon mapping algorithm.

7.4.1 Photon Tracing

In the presence of participating media, when a photon is emitted or reflected off a surface, it travels through the medium for some distance until it is either scattered or absorbed. As we saw in Chapter 4, the transmittance T_r describes the reduction of radiance as it travels through the medium due to absorption and out-scattering. Photon tracing can be augmented to propagate and store photons within the participating medium by probabilistically choosing the distance to the next interaction. The photon is advanced through the medium by this distance, at which point an interaction with the medium is simulated. At each interaction location, the photon is stored in a volume photon map.

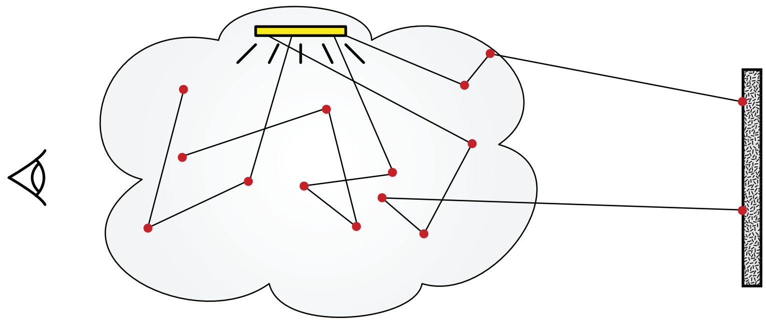

Figure 7.2: Photon mapping starts by emitting photons from light sources. In the presence of participating media, photons are stored not only on surfaces, but also within the medium.

Sampling Propagation Distance

Recall from Chapter 4 that the transmittance T_r actually gives the *probability* that a photon can travel unobstructed between two points in a medium. The propagation distance should therefore be computed by importance sampling the transmittance term. Given a random number $\xi \in (0,1]$, this can be accomplished with the following expression for d:

$$d(\mathbf{x}_p) = -\frac{\log(\xi)}{\bar{\sigma}_t(\mathbf{x}_p)}, \tag{7.6}$$

where the corresponding probability density is

$$pdf(d) = e^{-\bar{\sigma}_t(\mathbf{x}_p)d}. \tag{7.7}$$

In defining the above two equations, we introduced the term $\bar{\sigma}_t(\mathbf{x}_p)$. We have the freedom to define this parameter as any positive value; however, our goal is to use a value which results in a distance distribution proportional to the transmittance term T_r.

Homogeneous Media. For homogeneous media, if we set $\bar{\sigma}_t(\mathbf{x}_p)$ to the medium's extinction coefficient, $\bar{\sigma}_t(\mathbf{x}_p) = \sigma_t$, then the PDF above will be exactly proportional to the transmittance.

Heterogeneous Media. Distributing the distance according to transmittance is much more difficult in heterogeneous media and in general can only be approximated. The optimal solution would set $\bar{\sigma}_t(\mathbf{x}_p)$ to the mean extinction coefficient between $\mathbf{x}_p$ and the next interaction event

Algorithm 7.3: VolumetricPhotonTracing($\mathbf{x}_p$, $\vec{\omega}_p$, Φ_p)

$d = -\frac{\log(\xi)}{\bar{\sigma}_t(\mathbf{x}_p)}$;
$pdf(d) = e^{-\bar{\sigma}_t(\mathbf{x}_p)d}$;
while *no surface between* $\mathbf{x}_p$ *and* $\mathbf{x}_p + d\vec{\omega}_p$ **do**
  $\Phi_p \mathrel{*}= e^{-\sigma_t(\mathbf{x}_p)d} / pdf(d)$;
  $\mathbf{x}_p \mathrel{+}= d\vec{\omega}_p$;
  StorePhoton($\mathbf{x}_p$, $\vec{\omega}_p$, Φ_p);
  if $\xi < \frac{\sigma_s}{\sigma_t}$ **then**
    return;
  $\vec{\omega}_p$ = ComputeScatteredDirection($\mathbf{x}_p$, $\vec{\omega}_p$);
  $d = -\frac{\log(\xi)}{\bar{\sigma}_t(\mathbf{x}_p)}$;
  $pdf(d) = e^{-\bar{\sigma}_t(\mathbf{x}_p)d}$;
return;

at $\mathbf{x}_p + d\vec{\omega}$. Clearly this is not a practical approach since it requires knowledge of the distance d, which we are trying to compute in the first place! One approximate solution is to set the parameter to the extinction coefficient at the current scattering event, $\bar{\sigma}_t(\mathbf{x}_p) = \sigma_t(\mathbf{x}_p)$. In highly heterogeneous media this can lead to fairly erratic distance values, and care must be taken to ensure that $\bar{\sigma}_t > 0$. Another option is to compute the average extinction coefficient over the entire medium and use this as the value of $\bar{\sigma}_t(\mathbf{x}_p)$ for all $\mathbf{x}_p$. A more accurate, but potentially costly, approach is to choose the random value ξ and then incrementally compute T_r by taking small steps along $\vec{\omega}$ using ray marching. The next scattering event will occur where $T_r = \xi$.

Volumetric Scattering

At an interaction location, the photon can be either absorbed or scattered, and Russian roulette is used to choose between these two events. The ratio between the absorption and extinction coefficients describes the probability of absorption:

$$Pr\{\text{absorb}\} = \frac{\sigma_a}{\sigma_t}. \tag{7.8}$$

If the photon is not absorbed, it will be scattered into another direction through the medium. This direction can be computed by importance sampling the phase function.

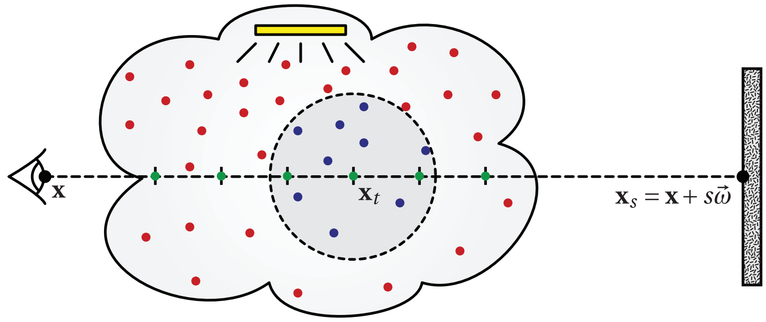

Figure 7.3: Volumetric photon mapping uses ray marching to accumulate in-scattered radiance along the length of an eye ray through the medium. At each discrete sample point the nearest photons are gathered, and the in-scattered radiance is approximated using the volumetric radiance estimate.

The volumetric photon tracing process is illustrated in Figure 7.2 and summarized in Algorithm 7.3.

7.4.2 Ray Marching and the Volumetric Radiance Estimate

During rendering, volumetric photon mapping uses ray marching (Equation 5.1) to numerically integrate the volume rendering equation for radiance seen directly by the observer:

$$L(\mathbf{x}\leftarrow\vec{\omega}) \approx T_r(\mathbf{x}\leftrightarrow\mathbf{x}_s)L(\mathbf{x}_s\rightarrow-\vec{\omega}) + \left(\sum_{t=0}^{S-1} T_r(\mathbf{x}\leftrightarrow\mathbf{x}_t)\sigma_s(\mathbf{x}_t)L_i(\mathbf{x}_t\rightarrow-\vec{\omega})\Delta_t\right). \tag{7.9}$$

This is illustrated in Figure 7.3. The most expensive part to compute in Equation 7.9 is the in-scattered radiance L_i, because it involves accounting for all light arriving at each point $\mathbf{x}_t$ along the ray from any other point in the scene. Instead of computing these values independently for each location, volumetric photon mapping gains efficiency by reusing the computation performed during the photon tracing stage. The in-scattered radiance is approximated using the *volumetric radiance estimate* by gathering photons within a small spherical neighborhood of radius r around each sample location $\mathbf{x}_t$,

$$L_i(\mathbf{x}_t,\vec{\omega}) \approx \sum_{p=1}^{k} p(\mathbf{x}_t,\vec{\omega}\leftrightarrow\vec{\omega}_p)\frac{\Phi_p(\mathbf{x}\rightarrow-\vec{\omega}_p)}{\mathcal{V}(\mathbf{x})}, \tag{7.10}$$

where $\mathcal{V}(\mathbf{x}) = \frac{4}{3}\pi d_k(\mathbf{x})^3$ is the volume of the sphere containing the nearest k photons. This is known as the *volume radiance estimate* and is illustrated in Figure 7.1.