

Progressive Photon Mapping

吴克文 梁家硕

2017 年 5 月

综述

本作业参考了三篇论文 [1][3][4]，以及书 [2] 和 Stanford CS148 课件，综合效果和实现难度进行了调整，删去了繁琐的细节调整和性能优化部分。

BRDF 参数及代码来自网站<http://www.merl.com/brdf/>

综述

本作业参考了三篇论文 [1][3][4]，以及书 [2] 和 Stanford CS148 课件，综合效果和实现难度进行了调整，删去了繁琐的细节调整和性能优化部分。

BRDF 参数及代码来自网站<http://www.merl.com/brdf/>

更多技术细节详见 Equestrotopia.pdf 和 src 文件夹中的代码以及 Github 仓库<https://github.com/liangjs/Equestrotopia>

① PPM vs PM

② 效果展示

③ 环境与使用

④ 总流程

⑤ 核心算法

Lighting Equation

Ray Tracing Pass

Photon Tracing Pass

Progressive Updation

⑥ 阶段性效果图比较

⑦ 参考文献

Progressive
Photon
Mapping

吴克文 梁家硕

PPM vs PM

效果展示

环境和使用

总流程

核心算法

Lighting
Equation

Ray Tracing
Pass

Photon Tracing
Pass

Progressive
Updation

阶段性效果图
比较

参考文献

Thanks

Section 1

PPM vs PM

Why Progressive Photon Mapping

Photon Mapping 作为全局光照领域的主流算法，以其高效率，能处理多种光照效果等特点，一直受到广泛的关注。

Why Progressive Photon Mapping

Photon Mapping 作为全局光照领域的主流算法，以其高效率，能处理多种光照效果等特点，一直受到广泛的关注。然而，Photon Mapping 算法的一个主要问题在于，使用光子进行光能估计的过程引入了偏差。理论上，要完全消除偏差，需要存储无穷的光子，这从计算机存储角度来看是不可接受的。

Why Progressive Photon Mapping

Photon Mapping 作为全局光照领域的主流算法，以其高效率，能处理多种光照效果等特点，一直受到广泛的关注。

然而，Photon Mapping 算法的一个主要问题在于，使用光子进行光能估计的过程引入了偏差。理论上，要完全消除偏差，需要存储无穷的光子，这从计算机存储角度来看是不可接受的。

为此，Toshiya Hachisuka 提出了 Progressive Photon Mapping(又称渐进式光子映射)，采用多遍的绘制流程，通过不断向场景中发射光子达到不断减小偏差的目的，亦解决了 Photon Mapping 的存储问题。

Progressive
Photon
Mapping

吴克文 梁家硕

PPM vs PM

效果展示

环境与使用

总流程

核心算法

Lighting
Equation

Ray Tracing
Pass

Photon Tracing
Pass

Progressive
Updation

阶段性效果图
比较

参考文献

Thanks

Section 2

效果展示

Progressive
Photon
Mapping

吴克文 梁家硕

PPM vs PM

效果展示

环境和使用

总流程

核心算法

Lighting
Equation

Ray Tracing
Pass

Photon Tracing
Pass

Progressive
Updation

阶段性效果图
比较

参考文献

Thanks

环境

Arch Linux x86_64 Linux 4.10.13-1-ARCH

gcc (GCC) 6.3.1 20170306

cmake version 3.8.0

GNU Make 4.2.1

OpenGL version: 3.0 Mesa 17.0.5

环境

Arch Linux x86_64 Linux 4.10.13-1-ARCH
 gcc (GCC) 6.3.1 20170306
 cmake version 3.8.0
 GNU Make 4.2.1
 OpenGL version: 3.0 Mesa 17.0.5

使用

```
$ cmake .
$ make
$ ./updation
```

Section 4

总流程

Algorithm 1: Progressive Photon Mapping

Input: source.obj material.mtl

Output: output.png

```

1 Read model and store material info.;
2 Build model KD-Tree based on model info in .obj;
3 Perform Ray Tracing Pass to restore hitpoints;
4 for iterations do
5     Perform Photon Tracing Pass;
6     Build photon KD-Tree with photon map;
7     for hitpoints do
8         Find photons near the hitpoint;
9         Accumulate their impact on the hitpoint;
10        Update hitpoint info.;
11    end
12 end
13 Generate output.png using hitpoints' info.;
```

Progressive
Photon
Mapping

吴克文 梁家硕

PPM vs PM

效果展示

环境和使用

总流程

核心算法

Lighting
Equation

Ray Tracing
Pass

Photon Tracing
Pass

Progressive
Updation

阶段性效果图
比较

参考文献

Thanks

Subsection 1

Lighting Equation

BRDF

BRDF(双向反射分布函数), 全称为 Bidirectional Reflectance Distribution Function, 用来定义给定入射方向上的辐射照度如何影响给定出射方向上的辐射率。更笼统地说, 它描述了入射光线经过某个表面反射后在各个出射方向上的分布效果。

BRDF

BRDF(双向反射分布函数), 全称为 Bidirectional Reflectance Distribution Function, 用来定义给定入射方向上的辐射照度如何影响给定出射方向上的辐射率。更笼统地说, 它描述了入射光线经过某个表面反射后在各个出射方向上的分布效果。

光照方程

$$L_o(x, \omega_o) = L_e(x, \omega_o) + \int_{\Omega} BRDF(x, \omega_i, \omega_o) L_i(x, \omega_i) (\omega_i \cdot n) d\omega_i \quad (1)$$

其中, L_e 为直接光照, x 为空间坐标, n 为平面法向量, ω_o, ω_i 分别为出射和入射方向。

Progressive
Photon
Mapping

吴克文 梁家硕

PPM vs PM

效果展示

环境与使用

总流程

核心算法

Lighting
Equation

Ray Tracing
Pass

Photon Tracing
Pass

Progressive
Updation

阶段性效果图
比较

参考文献

Thanks

Subsection 2

Ray Tracing Pass

光线追踪

从观察点出发，通过光线追踪来获得可见点 (hitpoints)，同时计算直接光照的贡献。

光线追踪

从观察点出发，通过光线追踪来获得可见点 (hitpoints)，同时计算直接光照的贡献。

注

在镜面较多的场景中可用反（折）射次数作为阈值强制结束 Ray Tracing Pass。

Algorithm 2: Ray Tracing Pass

Input: Camera, Lights, Width, Height, Scene

Output: Hitpoints, Pre-image

```
1 Initialize pre-image(width,height);
2 Initialize hitpoint list;
3 for i in range(0,width) do
4     for j in range(0,height) do
5         Initialize ray with (i,j) and camera place ;
6         Get the intersection of ray and scene;
7         while intersection is specular do
8             Determine whether it is reflected or refracted;
9             Change the direction of ray;
10            Re-get intersection;
11        end
12        Accumulate direct illumination;
13        Store hitpoint;
14    end
15 end
16 Return hitpoint list and pre-image;
```

Progressive
Photon
Mapping

吴克文 梁家硕

PPM vs PM

效果展示

环境与使用

总流程

核心算法

Lighting
Equation

Ray Tracing
Pass

**Photon Tracing
Pass**

Progressive
Updation

阶段性效果图
比较

参考文献

Thanks

Subsection 3

Photon Tracing Pass

光子追踪

每轮 Photon Tracing Pass，从光源随机方向发射一批光子，追踪每个光子的运动轨迹，考虑到效率，将光子能量的衰减用随机被物体表面吸收（或达到折反射阈值）来控制，这样每个光子的能量即为定值，折反射仅改变其颜色向量（通过 BRDF 计算）。

光子追踪

每轮 Photon Tracing Pass，从光源随机方向发射一批光子，追踪每个光子的运动轨迹，考虑到效率，将光子能量的衰减用随机被物体表面吸收（或达到折反射阈值）来控制，这样每个光子的能量即为定值，折反射仅改变其颜色向量（通过 BRDF 计算）。

注

由于直接光源已在 Ray Tracing Pass 计算过，故每个光子与场景的第一个交点不必计入 photon map。

Algorithm 3: Photon Tracing Pass

Input: Lights, Scene

Output: Photon map

```
1 Initialize photon map;
2 for number of photons do
3   Rand its initial direction;
4   while photon not absorbed and not up to limitation do
5     Get its intersection with scene;
6     Store intersection in photon map;
7     if photon not absorbed then
8       Determine whether it is reflected or refracted;
9       Update its direction;
10    end
11  end
12 end
13 Return photon map;
```

Progressive
Photon
Mapping

吴克文 梁家硕

PPM vs PM

效果展示

环境与使用

总流程

核心算法

Lighting
Equation

Ray Tracing
Pass

Photon Tracing
Pass

**Progressive
Updation**

阶段性效果图
比较

参考文献

Thanks

Subsection 4

Progressive Updation

更新模型

结束 Photon Tracing Pass 后，需要枚举每个 hitpoint，同时统计其半径 R 内光子对其亮度影响。

更新模型

结束 Photon Tracing Pass 后, 需要枚举每个 hitpoint, 同时统计其半径 R 内光子对其亮度影响。

推导

记 $N(x)$ 为上轮后在 hitpoint x 半径 $R(x)$ 内的光子数, $M(x)$ 为本次新增光子数, 同时 $\hat{N}(x), \hat{R}(x)$ 分别为新累计光子数和半径, 则有如下更新,

$$\hat{N}(x) = N(x) + \alpha M(x) \quad (2)$$

$$\hat{R}(x) = R(x) \sqrt{\frac{N(x) + \alpha M(x)}{N(x) + M(x)}} \quad (3)$$

推导

记 $\tau_N(x, \omega)$ 和 $\tau_M(x, \omega)$ 为在 x 处，入射光方向为 ω 的前光强和新增光强（未乘 BRDF 系数），则有

$$\tau_{\hat{N}}(x, \omega) = (\tau_N(x, \omega) + \tau_M(x, \omega)) \frac{N(x) + \alpha M(x)}{N(x) + M(x)} \quad (4)$$

其中， $\alpha \in (0, 1)$ 是一常数。

推导

再记总发射光子数为 $N_{emitted}$ ， ϕ 为光子光强，则最终辐照率表达式为，

$$L(x, \omega) = \int_{2\pi} BRDF(x, \omega, \omega') L(x, \omega') (n \cdot \omega') (d\omega') \quad (5)$$

$$\approx \frac{1}{\Delta A} \sum_{p=1}^n BRDF(x, \omega, \omega') \Delta\phi_p(x_p, \omega_p) \quad (6)$$

$$= \frac{1}{\pi R(x)^2} \frac{\tau(x, \omega)}{N_{emitted}} \quad (7)$$

Progressive
Photon
Mapping

吴克文 梁家硕

PPM vs PM

效果展示

环境和使用

总流程

核心算法

Lighting
Equation

Ray Tracing
Pass

Photon Tracing
Pass

Progressive
Updation

阶段性效果图
比较

参考文献

Thanks

Section 6

阶段性效果图比较

Progressive
Photon
Mapping

吴克文 梁家硕

PPM vs PM

效果展示

环境和使用

总流程

核心算法

Lighting
Equation

Ray Tracing
Pass

Photon Tracing
Pass

Progressive
Updation

阶段性效果图
比较

参考文献

Thanks



Toshiya Hachisuka, Shinji Ogaki, and Henrik Wann Jensen.

Progressive photon mapping.

ACM Transactions on Graphics (TOG), 27(5):130, 2008.



Henrik Wann Jensen.

Realistic image synthesis using photon mapping, volume 364.

Ak Peters Natick, 2001.



Ben Spencer and Mark W Jones.

Progressive photon relaxation.

ACM Transactions on Graphics (TOG), 32(1):7, 2013.



李睿, 陈彦云, and 刘学慧.

基于自适应光子发射的渐进式光子映射.

计算机工程与设计, 33(1):219–223, 2012.

Thanks!