Fotonové mapy

Leonid Buneev

21. 01. 2012

Popis algoritmu

Photon mapping – algoritmus, který, stejně jako path tracing a bidirectional path tracing, vyřeší zobrazovací rovnice, ale podstatně jiným způsobem. Tenhle algoritmus už není nestranný (průměr velkého počtu zobrazování nekonverguje ke správnému řešení zobr. rovnice), ale je konzistentní (výsledek konverguje při zvětšování počtu fotonů). Při stejné kvalitě zobrazování většinou je mnohem rychlejší, než algoritmy, založené na metodách Monte-Carlo, a umožňuje spočítat důležité efekty, jako:

- Caustics (Kaustiky)
- Color bleeding (Difuzní odrazy)
- Subsurface scattering (rozptyl světla pod povrchem tělesa)

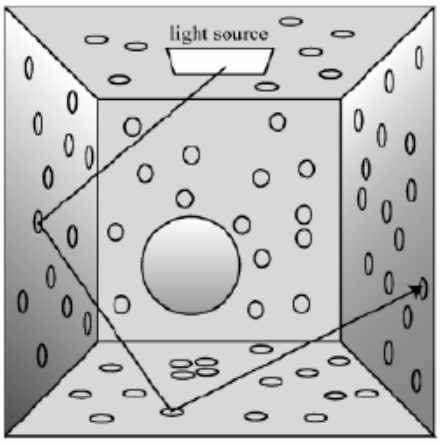
Průběh algoritmu

Photon mapping se skládá ze dvou částí:

1. Rozmístění fotonů
   V podstatě 1. fáze je light tracing – sledujeme cesty světla (a la fotony), emitovaného ze světelného zdroje, do scény, a výsledky (osvětlení scény v dosažených bodech) kešujeme (ukládáme ji do fotonové mapy)

2. Rendering
   Sledujeme cesty od kamery – jako path tracing, ovšem místo rekurze používáme již spočítané hodnoty z fotonové mapy

1. Fáze: Rozmístění fotonů



1. Emise fotonů

Chceme, aby všechny emitované fotony nesly stejný tok – v tomhle případě by variance odhadu světla z fotonové mapy byla nižší. Pro emise každého fotonu nejdříve vybereme náhodně světelný zdroj (s pravděpodobnosti, úměrnou celkovému toku zdroje). Pak vybereme počátek fotonu: v případě bodového zdroje světla je to triviální – pozice zdroje, pro plošné zdroje – náhodný bod na ploše světelného zdroje. Pak

vybereme směr emitovaného fotonu – taky náhodně, s pravděpodobností úměrnou emisní distribuční funkce světla.

2. Sledování fotonů

Probíhá podobným způsobem jako light tracing. Když ale foton se narazí na plochu, děláme následující věcí:

1.) Uložíme foton do fotonové mapy – „foton" je trojice (pozice, příchozí směr, tok)
2.) Vygenerujeme směr odraženého paprsku – pomocí BRDF importance samplingu
3.) Aktualizujeme tok fotonu (předběžný)

$$\Phi_{p,j+1}^{tentative} = \Phi_{p,j} \frac{f_r(\mathbf{x}, \omega_o \rightarrow \omega_i)\left|\cos\theta_o\right|}{p(\omega_o)}$$

4.) Spočítáme pravděpodobnost přežití fotonu pomocí ruské rulety a, pokud přežije, pokračujeme dal.

$$q_{p,j+1} = \min\left\{1, \frac{\mathrm{Lum}[\Phi_{p,j+1}^{tentative}]}{\mathrm{Lum}[\Phi_{p,j}]}\right\}$$

Pravděpodobnost přežití

$$\Phi_{p,j+1} = \frac{\Phi_{p,j+1}^{tentative}}{q_{p,j+1}}$$

Výsledný tok fotonu při přežití

Tenhle postup je dobrý tím, že zachovává luminanci fotonů, což je dobrý pro budoucí rendering. Při lomu světla taky není třeba měnit tok fotonů – fotony nenesou žádnou radiancí.

3. Ukládání fotonu do fotonové mapy

Pro většinu scén stačí $10^5 - 10^7$ fotonů. Pokaždé, když foton se narazí na difuzní (nebo mírně lesklou) složku, je potřeba uložit foton do mapy (i při absorpci taky). Ovšem, když narazíme na zrcadlovou plochu, jenom pokračujeme sledování fotonu – nemá smysl ukládat světlo do mapy, protože při sledovaní cest od kamery pří narazení na zrcadlovou plochu žádná interpolace osvětlení okolních bodů nedává smysl.

Během rozmísťování stavíme pouze lineární seznam fotonů, pak, po rozmístění, stavíme nad tím kD-strom, který umožnuje rychlejší vyhledávání v budoucnosti. Foton je trojice (pozice, příchozí směr, tok):

- Pozice $x_p = (x, y, z)$
- Směr $\omega_p = (\theta, \phi)$ -> char[2]
- Energie (tok) $\Phi_p = (r, g, b)$ -> RGBE: char[4]

Pro vypočet kaustik je vhodný postavit zvláštní fotonovou mapy (mapa kaustik), která bude podmnožinou původní (globální) mapy a obsahovat jenom nepřímé osvětlení.

Odhad radiance z fotonové mapy

Pro rendering potřebujeme rychlé hledat k nejbližších fotonu v nějakém bodě, proto ze seznamu fotonů vytvoříme kD-strom. Pak umíme odhadnout radiancí v daném bodě podle formuli:

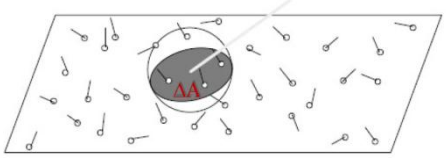
k .. počet fotonů v okolí bodu $\mathbf{x}$

směr incidence fotonu p

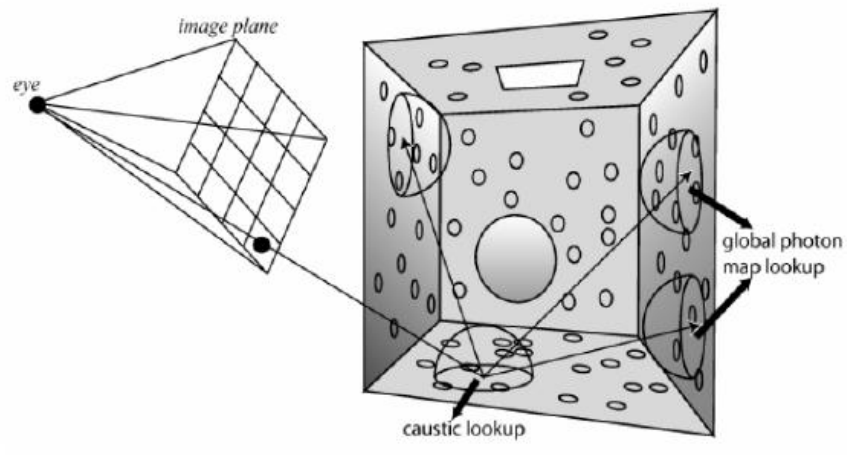
$$L_r(\mathbf{x}, \omega_o) \approx \sum_{p=1}^{k} f_r(\mathbf{x}, \omega_p, \omega_o) \frac{\Phi_p(\mathbf{x}, \omega_p)}{\Delta A}$$

$$\approx \frac{1}{\pi r^2} \sum_{p=1}^{k} f_r(\mathbf{x}, \omega_p, \omega_o) \Phi_p(\mathbf{x}, \omega_p)$$

obsah kruhové oblasti ΔA



2. Fáze: Rendering



Rendering probíha stejně jako ve klasickém path tracingu z kamery, s tím, že rekurze je nahrazená odhadem z fotonové mapy pro difuzní povrchy. Pro ideální zrcadlové povrchy pořád se používá klasická rekurze.

Začneme klasickou rovnicí pro vypočet odražené radianci:

$$L_r(\mathbf{x},\omega_o) = \int_\Omega L_i(\mathbf{x},\omega_i)\, f_r(\mathbf{x},\omega_i \rightarrow \omega_o)\cos\theta_i \,\mathrm{d}\omega_i$$

Příchozí radiance L_i rozdělime na tři částí - přímé osvětlení, kaustiky a nepřímé osvětlení:

$$L_i = L_{i,d} + L_{i,c} + L_{i,l}$$

BRDF f_r - na dvě - difuzní a zrcadlový odraz:

$$f_r = f_{r,D} + f_{r,S}$$

A každý případ spočítáme odpovídajícím způsobem:

	$f_{r,D}$	$f_{r,S}$
$L_{i,d}$	přímé osvětlení	ideální zrcadlové odrazy/lomy
$L_{i,c}$	kaustiky **(PM)**	
$L_{i,l}$	difúzní nepřímé **(FG +PM)**	

, kde PM je Photon Mapping a FG je Final Gathering (bude vysvětlen později)

Přímé osvětlení a zrcadlové odrazy se počítají jako obyčejně – pomocí vzorkování světel a stínových paprsku resp. pomocí určitých sekundárních paprsků. Kaustiky se počítají odhadem radiance z fotonové mapy kaustik. Nepřímé osvětlení se počítá pomocí Final Gathering, který spočívá v tom, že místo odhady v daném bodě z fotonové mapy pošleme několik paprsku z tohoto bodu do scény, a průsečíky těch paprsku se scénou už vyhodnotíme odhadem z globální fotonové mapy. FG je potřeba používat z toho důvodu, že informace z fotonové mapy je příliš nepřesná, a použitím mapy až pro sekundární paprsky nepřesnosti se „zprůměrují“.

přímé použití

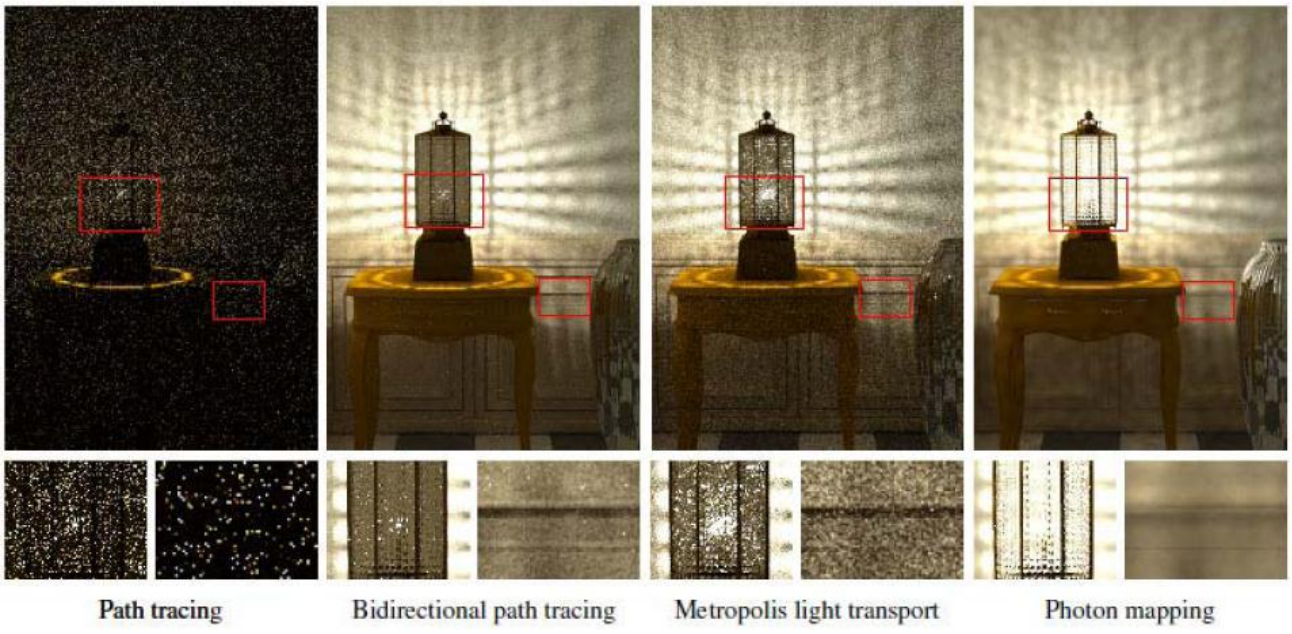
final gathering

Pro vypočet kaustik obvykle není třeba používat FG, protože v těchto bodech je dostatečná hustota fotonu pro pěkný odhad.

Existují možnosti zrychlit Final Gathering. Jednou z možností je irradiance caching, která používá ten fakt, že osvětlení difuzní plochy se mění pozvolna a plynule. Druhou možností je předvýpočet radiance na pozicí fotonu – po rozmístění všech fotonů vybereme podmnožinu fotonů, ve které pro každý foton spočteme odhad radiancí z fotonové mapy, a ty výsledky uložíme do separátního kD-stromu. Tím pádem během renderingu nebudeme muset hledat množinu nejbližších fotonu z fotonové mapy – stačí najit nejbližší přepočítanou radiancí. Ale tenhle metod funguje pouze pro dotaz do fotonové mapy na difuzní ploše.

Vlastnosti algoritmu

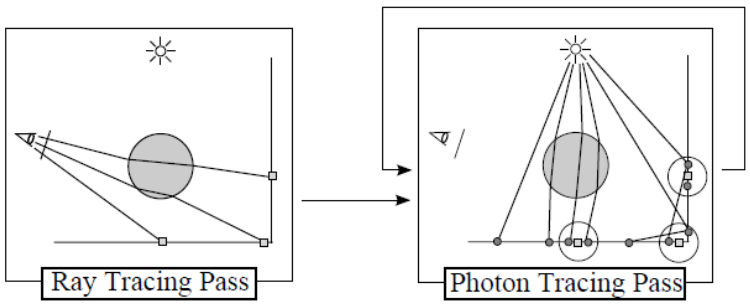
Už víme, že fotonové mapy jsou vynikající pro zobrazování kaustik. V podstatě je dobrý pro zobrazování světla, které prochází cestou specular-diffuse-specular (tzv. SDS path) – žádné Monte-Carlo algoritmy takové světlo zobrazit neumí. Příkladem SDS-cest mohou byt kaustiky na dně bazénu, nebo světlo, emitované malým zdrojem světla, který nachází v nějakém skleněném obalu.



Path tracing | Bidirectional path tracing | Metropolis light transport | Photon mapping

Ovšem tenhle algoritmus má i svoje nevýhody. Chová se docela divně na lesklých plochách – sekundární paprsky, které vytváříme během final gatheringu, často dopadnou do stejného místa ve scéně, a proto vidíme jakoby „odraz“ fotonové mapy se všemi příslušnými ji nepřesnosti.

Teoreticky taky není vůbec ideální – výsledek není nestranný, obsahuje systematickou chybu, a proto nekonečné zprůměrování renderovaných obrázků nedává dobrý výsledek. Ale výsledek je konzistentní – při nekonečném počtu fotonu konverguje ke správnému řešení. Kvůli omezené velikosti paměti toho nelze dosáhnout, ale řešením tohoto problému je trochu upravený algoritmus – progressive photon mapping. Tento algoritmus nejdříve vrhá paprsky do každého pixelu ve scéně jako v Ray Tracingu, pamatuje tyhle body, a pak opakovaně spočítá fotonové mapy, které používá pro počítání radiance v nalezených průsečících paprsků se scénou. Při tom v každém kroku zmenšíme poloměr hledání nejbližších fotonu, aby celkový bias a rozptyl šli k nule.



Dalším vylepšením fotonových map jsou robustní fotonové mapy. Zatím jsme pří vrhání paprsku z kamery pro difuzní povrchy hned počítali radiance z fotonové mapy, a pro lesklé povrhy pokračovali cestu. Robustní fotonové mapy počítají radiance na všech vrcholech cesty od kamery, a kombinují spočítané hodnoty pomocí MIS.

Final gather
(dotaz do PM na druhém vrcholu cesty od kamery)

Direct visualization
(dotaz do PM na prvním vrcholu cesty od kamery)

Naše řešení
(dotaz do PM na všech vrcholech cesty do kamery + kombinace pomocí MIS)

Posledním zatím vymyšleným vylepšením je použití fotonových map jenom jako další možnou vzorkovací strategie pro Bidirectional Path Tracing.

PPM

Náš výsledek (Vorba): Robustní PPM

Náš výsledek (Georgiev): Robustní PPM + BDPT