Vzorkovací algoritmy v počítačové grafice

Josef Pelikán*
MFF UK Praha

Se zdokonalováním algoritmů pro realistickou syntézu obrazu stoupá význam nestandardních vzorkovacích metod. V tomto příspěvku uvedeme přehled nejpoužívanějších vzorkovacích algoritmů s porovnáním jejich efektivity a kvality.

1 Úvod

Při vytváření obrazu se v počítačové grafice často používá rastrově-orientovaný přístup: pro každý bod v pravidelném diskrétním rastru se spočítá barva a výsledný obrázek je pak tvořen maticí těchto bodů (bodům říkáme **pixely** z anglického "picture element"). Takto získaný obraz vyhovuje i výstupnímu zobrazovacímu zařízení, které bývá téměř vždy rastrové.

Nejjednodušší metody pracují tak, že určí barvu v jediném místě každého pixelu, např. v jeho středu. Velmi snadno však bývá obraz znehodnocen efekty, které se souhrnně nazývají **"alias"** a mají původ v nedostatečném vzorkování obrazu a jeho interferenci s pravidelnou sítí pixelů. Jako známý příklad tohoto jevu můžeme uvést zubaté kreslení šikmých úseček na obrazovce. Můžeme sice zvětšovat použité rozlišení, ale jsme většinou omezeni technikou a jak za chvíli poznáme, toto řešení vůbec neodstraňuje princip našich potíží.

Představme si na chvíli obrázek plotu jdoucího do nekonečna nakresleného v perspektivní projekci. V blízkosti úběžníku se plaňky plotu střídají "nekonečně rychle" - a to je právě důvod, proč pouhým zvětšováním rozlišení nemůžeme dosáhnout dobrého výsledku, vždy totiž dojde v některém místě k interferenci.

Než si ukážeme lepší řešení, musíme zavést přesněji některé pojmy: originální obraz budeme reprezentovat tzv. **obrazovou funkcí** $f(x,y)$, spojitou a se spojitým oborem hodnot (např. barva v prostoru RGB nebo úroveň jasu). Výstupní obraz je diskrétní - budeme předpokládat čtvercový rastr s jednotkovou vzdáleností pixelů. Hodnotu pixelu o souřadnicích $[x,y]$ označíme $I(x,y)$. Nyní můžeme chápat pixel jako plochu a použít k výpočtu jeho barvy střední hodnotu

$$I(X,Y) = \int_X^{X+1} \int_Y^{Y+1} f(x,y)\ dy\ dx \tag{1}$$

nebo v obecnějším případě váženou střední hodnotu

$$I(X,Y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x,y)\ g(x-X, y-Y)\ dy\ dx \tag{2}$$

*KSVI MFF UK, Malostranské nám. 25, 11800 Praha 1, e-mail: pepca@sun1.ms.mff.cuni.cz

s pomocí tzv. **filtru** g (vztah (1) je speciálním případem rovnice (2) s filtrem $g(u,v) \equiv 1$ na intervalu $\langle 0,1\rangle \times \langle 0,1\rangle$ a $g(u,v) \equiv 0$ jinde). Metoda střední hodnoty již může odstranit z obrazu "alias", předpokládá však výpočet integrálu (1) nebo (2), který je často analyticky neřešitelný.

Zaměříme se nyní na populární metodou **"rekurzivního sledování paprsku"**, která slouží k realistickému zobrazování 3D scén. Zavedl ji roku 1979 Whitted (viz [Whitted80], [Glassner89]). K výpočtu funkce f používáme paprsek, jehož dráhu ve scéně zpětně sledujeme (počítáme s odrazem a lomem světla). Výsledkem sledování je barva obrazu $f(x,y)$ v jediném bodě, a tedy při posílání jednoho paprsku na pixel může vznikat "alias". Jedním pixelem proto vysíláme více paprsků a říkáme tomu **"vícenásobné vzorkování"** (supersampling). Označíme-li souřadnice jednotlivých vzorků $[x_i, y_i]$ pro $i = 1, 2, ..N$, můžeme jako aproximaci (1) použít vztah

$$I(X,Y) = \frac{1}{N} \sum_{i=1}^{N} f(x_i, y_i). \tag{3}$$

Analogicky bychom mohli aproximovat obecnější rovnost (2).

2 Vzorkovací algoritmy

Vzorkovacím algoritmem rozumíme postup, jak spočítat souřadnice N vzorkovacích bodů $[x_1, x_2, ..x_m]_i$, $i = 1, 2, ..N$ z požadované m-dimenzionální oblasti (nejčastěji to bývá pravoúhlý interval nebo koule). Vzorkovací body se potom používají při odhadu integrálu (1) podle vztahu (3). Aby byl odhad co nejlepší, požadujeme rovnoměrné rozložení vzorků v dané oblasti, úplná pravidelnost bývá však na závadu, protože způsobuje již zmiňované interference.

V dalších odstavcích si ukážeme několik základních vzorkovacích algoritmů; pro jednoduchost se omezíme na dvojrozměrný případ a pokrývání jednotkového čtverce $\langle 0,1\rangle \times \langle 0,1\rangle$. Většinu uvedených metod lze rozšířit do více dimenzí a případně i upravit pro jiné vzorkované oblasti.

U jednotlivých metod je v pseudokódu uveden postup výpočtu, funkce **Rand(a,b)** vrací náhodné číslo s rovnoměrným rozdělením v intervalu $\langle a, b\rangle$.

2.1 Pravidelné vzorkování

Vzorky umístíme do pravidelné čtvercové nebo obdélníkové sítě. Pravidelnost může způsobit nepříjemné interference.

```
for i = 0 to n_x − 1
    for j = 0 to n_y − 1
        k = n_x i + j + 1
        x_k = (i + 0.5)/n_x
        y_k = (j + 0.5)/n_y
```

kde n_x a n_y jsou taková celá čísla, aby $n_x n_y = N$.

2.2 Náhodné vzorkování

Vzorky jsou nezávislé a vybírají se náhodně z celého čtverce. Díky vlastnostem rovnoměrného rozdělení se mohou tvořit shluky a obraz je pak zatížen velkým šumem.

for $i = 1$ **to** N
    $x_i = \mathbf{Rand}(0, 1)$
    $y_i = \mathbf{Rand}(0, 1)$

2.3 Roztřesení (jittering)

Vzorkovaný čtverec je pokryt sadou stejných čtverců nebo obdélníků a v každém z nich se náhodně vybírá jeden vzorek. Tím jsou potlačeny shluky i úplná pravidelnost a zachováno rovnoměrné rozložení vzorků.

for $i = 0$ **to** $n_x - 1$
    for $j = 0$ **to** $n_y - 1$
        $k = n_x i + j + 1$
        $x_k = \mathbf{Rand}(i/n_x, (i + 1)/n_x)$
        $y_k = \mathbf{Rand}(j/n_y, (j + 1)/n_y)$

kde $n_x n_y = N$ jako u pravidelného vzorkování.

Někdy se vzorky nevybírají z celých čtverců (obdélníků), ale jen z menších oblastí. O **polovičním roztřesení** (semijittering) mluvíme tehdy, jestliže zmenšíme na polovinu délku stran čtverců (obdélníků):

for $i = 0$ **to** $n_x - 1$
    for $j = 0$ **to** $n_y - 1$
        $k = n_x i + j + 1$
        $x_k = \mathbf{Rand}((i + 0.25)/n_x, (i + 0.75)/n_x)$
        $y_k = \mathbf{Rand}((j + 0.25)/n_y, (j + 0.75)/n_y)$

2.4 Poissonovo diskové vzorkování

Toto vzorkování napodobuje rozdělení světločivných buněk na sítnici lidského oka. Buňky jsou rozloženy náhodně, ale žádné dvě se nemohou dostat k sobě blíž než na jistou vzdálenost **d**. Takové rozložení zaručuje dobré vysokofrekvenční vlastnosti (rovnoměrné ale nepravidelné rozložení, potlačení jakýchkoli shluků).

$i = 1$
while $i <= N$
    $x_i = \mathbf{Rand}(0, 1)$
    $y_i = \mathbf{Rand}(0, 1)$
    $OK = true$
    for $j = 1$ **to** $i - 1$
        if $(x_i - x_j)^2 + (y_i - y_j)^2 < d^2$ **then** $OK = false$; **break**
    if OK **then** $i = i + 1$

Generování Poissonova diskového rozdělení je málo efektivní a navíc je obtížné zvolit vhodnou velikost konstanty **d**. V následujícím odstavci je uvedena praktičtější modifikace tohoto rozdělení.

2.5 Sekvenční Poissonovo diskové vzorkování

Don Mitchell publikoval algoritmus generující vzorky se zvětšující se hustotou a podobným rozdělením jako má Poissonův disk ([Mitchell91]). Algoritmus má jediný celočíselný parametr **m**, který ovlivňuje kvalitu vzorku. Dobrých výsledků lze dosáhnout při volbě $m \geq 10$.

$x_1 = \mathbf{Rand}(0,1)$
$y_1 = \mathbf{Rand}(0,1)$
for $i = 2$ **to** N
    $d_{max} = 0$
    for $j = 1$ **to** $m \cdot i$
        $x = \mathbf{Rand}(0,1)$
        $y = \mathbf{Rand}(0,1)$
        $d = \min\{(x - x_k)^2 + (y - y_k)^2\}_{k=1..i-1}$
        if $d > d_{max}$ **then** $d_{max} = d;\ x_i = x;\ y_i = y$

2.6 Vzorkování "N věží"

Tento vzorkovací algoritmus je úspornou variantou roztřesení ([Cook84], Shirley90]). Ve vyšších dimenzích se totiž ukázalo, že pro dobrý odhad integrálu (1) není třeba počítat n^m vzorků (kde m je dimenze), ale při vhodném vzájemném přiřazení intervalů v jednotlivých rozměrech postačí n vzorků. Ve vyšších dimenzích se tomuto vzorkování říká **"nezávislé roztřesení"** (uncorrelated jittering).

for $i = 1$ **to** N
    $x_i = \mathbf{Rand}(i/N, (i+1)/N)$
    $y_i = \mathbf{Rand}(i/N, (i+1)/N)$
"náhodná permutace hodnot x_1 **až** x_N**"**

2.7 Adaptivní zjemňování

Při zobrazování scén v počítačové grafice se stává, že funkce $f(x, y)$ se podstatně mění pouze v některých částech pixelu. Proto je zbytečné počítat její hodnotu všude stejně hustě. Téměř všechny dosud uváděné algoritmy se dají upravit do adaptivní formy - nejprve se spočítá hrubý odhad s malým počtem vzorků a potom se další vzorky počítají pouze tam, kde "se něco děje".

Nyní si uvedeme příklad jednoduchého vzorkovacího algoritmu, který byl sestaven přímo v adaptivní podobě ([Glassner89]). Protože se v něm nevyužívá náhodné rozdělení, má stejné nevýhody jako pravidelné vzorkování. Algoritmus je popsán rekurzivně, funkce $Sample(x_1, y_1, x_2, y_2)$ vrací odhad integrálu (1) na čtverci $\langle x_1, x_2 \rangle \times \langle y_1, y_2 \rangle$.

Sample(x_1, y_1, x_2, y_2)
$x_m = (x_1 + x_2)/2;\ y_m = (y_1 + y_2)/2$
$a = f(x_1, y_1);\ b = f(x_2, y_1)$
$c = f(x_1, y_2);\ d = f(x_2, y_2);\ e = f(x_m, y_m)$
if $x_2 - x_1 < d_{min}$ **then return** $(a + b + c + d)/8 + e/2$
if $|a - e| > treshold$ **then** $a = Sample(x_1, y_1, x_m, y_m)$
else $a = (a + e)/2$
if $|b - e| > treshold$ **then** $b = Sample(x_m, y_1, x_2, y_m)$
else $b = (b + e)/2$
if $|c - e| > treshold$ **then** $c = Sample(x_1, y_m, x_m, y_2)$
else $c = (c + e)/2$
if $|d - e| > treshold$ **then** $d = Sample(x_m, y_m, x_2, y_2)$
else $d = (d + e)/2$
return $(a + b + c + d)/4$

3 Prostory s vyšší dimenzí

Cook, Porter a Carpenter publikovali rozšíření základního algoritmu sledování paprsku v roce 1984 ([Cook84], [Glassner89]). Metoda, které se začalo říkat **distribuované sledování paprsku**, dokáže postihnout efekty, jako je třeba **hloubka ostrosti**, **rozmazání pohybem**, **měkké odrazy a stíny**. Princip spočívá ve sledování více paprsků a jejich rozdělování v průběhu výpočtu.

Např. rozmazání pohybem docílíme vržením několika paprsků v různých časových okamžicích, některé z nich mohou zasáhnout rychle pohybující se těleso a jiné ne. Průměrováním výsledků dosáhneme toho, že je pohybující se těleso rozmazané. Hloubka ostrosti se zase simuluje vysláním svazku paprsků z různých míst objektivu pomyslného fotoaparátu, atd.

Společným znakem všech těchto vylepšení je zvětšení dimenze prostoru, ve kterém odhadujeme integrál (1). Při kombinaci několika metod pak můžeme dostat třeba devítinásobný integrál. Dva rozměry (x,y): vzorkování uvnitř pixelu, dva rozměry (u,v): povrch čočky, jeden rozměr (t): čas, dva rozměry (a,b): povrch plošného světelného zdroje a dva rozměry (α,β): rozptyl paprsků při odrazu.

$$I(X,Y) = \int_X^{X+1} \int_Y^{Y+1} \int\int\int\int\int\int\int f(x,y,u,v,t,a,b,\alpha,\beta)\, d\beta\, d\alpha\, db\, da\, dt\, dv\, du\, dy\, dx$$

Odhadujeme-li tento integrál, nemůžeme z praktických důvodů používat pravidelné nebo obyčejné roztřesené vzorkování, úplný počet vzorků pro pouhé dvě hodnoty v každé dimenzi by byl $2^9 = 512$! Náhodné vzorkování nebo nezávislé roztřesení jsou zde naopak efektivní.

4 Implementace

Poissonovo diskové vzorkování se při implementaci ukázalo jako nepraktické, proto byla použita jeho sekvenční varianta s dopředu spočítaným souborem vzorků. Zlepšení uvedené v [Měch93] bylo dále upraveno takto:

- sekvenčním algoritmem bylo spočítáno několik tisíc vzorků v jednotkovém kruhu (řádově hodiny CPU)
- při generování vzorků uvnitř jednoho pixelu se celý soubor vzorků otočil o náhodný úhel, posunul náhodně o ± 0.2 a pak se vybíraly pouze vzorky, které padly do čtverce $\langle 0,1\rangle \times \langle 0,1\rangle$

Toto vzorkování bude dále nazýváno "lokální Poissonův disk", protože je v každém pixelu použit stejný (i když pootočený a posunutý) soubor vzorků. Globální varianta byla implementována tak, že se dopředu spočítalo několik tisíc vzorků ve čtverci 8×8 pixelů a tento soubor se periodicky na ploše obrazu opakoval.

5 Výsledky

Popsané algoritmy byly testovány na třech zkušebních obrazech:

Všechny tři obrázky mají rozlišení 200×150 pixelů a rozsah odstínů šedi $\langle 0,1\rangle$. Obrazové funkce byly rozšířeny o další dvě proměnné tak, aby integrál (1) vycházel ve třech a čtyřech dimenzích pro všechny pixely roven 0.5. Vzorový obraz ve dvou dimenzích byl získán roztřeseným vzorkováním s 2500 vzorky/pixel. Popis testovacích obrazů:

- **Šachovnice:** do nekonečna ubíhající šachovnice, převládají vodorovné a svislé linie. $f(x,y,t,z) = Odd(\lfloor (x-x_0)/y+16t\rfloor + \lfloor 1/y+xz\pi^{-2}\rfloor)$, kde $Odd(x)$ je funkce vracející 1 pro lichý argument a 0 pro sudý.
- **Šachovnice 45:** šachovnice otočená o 45^o, převládají v ní šikmé linie. Obrazová funkce $f(x,y,t,z) = Odd(\lfloor (x-x_0+1)/y + xz\pi^{-2}\rfloor + \lfloor (x-x_0-1)/y+16t\rfloor)$.

- **Vlny:** rychle se zahušťující soustředné kruhy, tento obrázek obsahuje velmi vysoké frekvence. $f(x, y, t, z) = Odd(\lfloor x^2 + y^2 + xyz^2 + 8t \rfloor)$.

Osm testovaných vzorkovacích algoritmů bylo vyzkoušeno na všechny tři funkce při odhadování dvojitého, trojitého a čtyřnásobného integrálu. Ve dvojrozměrném případě byly výsledky porovnávány se vzorovým obrazem (viz výše), v ostatních případech posloužila pro srovnání analyticky spočítaná hodnota 0.5. Ve třech následujících tabulkách jsou uvedeny průměrné absolutní chyby jednoho pixelu obrazu v tisícinách rozsahu škály. První tabulka obsahuje chyby změřené při 2D vzorkování:

	Šachovnice		Šachovnice 45		Vlny	
vzorkování	16/pix	64/pix	16/pix	64/pix	16/pix	64/pix
pravidelné	26.3	13.1	24.6	9.3	**35.2**	**11.9**
náhodné	30.5	15.4	35.8	17.8	93.9	46.6
roztřesení	20.7	8.2	24.8	9.8	57.2	20.4
poloviční roztřesení	21.6	9.4	**22.8**	**9.1**	43.7	15.7
nezávislé roztřesení	**16.6**	**6.8**	23.2	10.9	69.7	33.7
lokální Poissonův disk	19.9	8.5	23.9	10.2	51.1	21.3
globální Poissonův disk	20.7	8.6	25.0	10.8	53.0	22.0
adaptivní	19.1	10.3	30.0	22.8	49.3	42.8

V další tabulce jsou uvedeny chyby naměřené při 3D vzorkování:

	Šachovnice		Šachovnice 45		Vlny	
vzorkování	8/pix	64/pix	8/pix	64/pix	8/pix	64/pix
pravidelné	399.3	222.3	400.7	384.5	134.9	104.4
náhodné	**136.4**	49.8	137.4	49.4	136.5	49.9
roztřesení	136.9	49.0	**136.1**	49.7	137.0	49.9
poloviční roztřesení	136.5	50.0	136.7	49.7	136.4	78.2
nezávislé roztřesení	137.0	**27.5**	136.4	**28.6**	135.8	**47.1**
lokální Poissonův disk	170.9	27.7	169.5	29.1	**130.3**	47.9

Třetí tabulka obsahuje chyby naměřené při 4D vzorkování:

	Šachovnice		Šachovnice 45		Vlny	
vzorkování	16/pix	256/pix	16/pix	256/pix	16/pix	256/pix
pravidelné	395.0	139.3	219.3	135.0	**96.2**	47.0
náhodné	98.1	25.0	98.3	25.0	98.2	24.8
roztřesení	**97.8**	25.0	97.9	24.9	97.9	25.0
poloviční roztřesení	98.5	25.0	99.1	24.8	98.4	33.7
nezávislé roztřesení	98.2	**24.5**	**97.8**	**24.4**	98.0	**24.4**
lokální Poissonův disk	99.8	27.0	99.4	27.0	98.0	25.2

Aby bylo možno porovnat relativní kvalitu vzorkovacích algoritmů, byly všechny srovnatelné výsledky seřazeny podle velikosti a pro jednotlivé algoritmy spočteno průměrné

pořadí. Poslední tabulka zachycuje výsledky tohoto porovnání:

Celkové pořadí		**Pořadí ve 2D**		**Pořadí v 3D a 4D**	
1. nezávislé	2.5	1. polo-roztřesení	2.8	1. nezávislé	1.9
2. roztřesení	3.2	2. lokální Poisson	3.5	2. roztřesení	2.9
3. polo-roztřesení	3.6	3. pravidelné	3.7	3. náhodné	3.3
4. lokální Poisson	3.6	4. nezávislé	3.8	4. lokální Poisson	3.7
5. pravidelné	4.7	5. roztřesení	3.9	5. polo-roztřesení	4.0
6. náhodné	4.8	6. globální Poisson	4.9	6. pravidelné	5.3
7. globální Poisson	4.9	7. adaptivní	5.7		
8. adaptivní	5.7	8. náhodné	7.7		

Z výsledků vyplývá, že nelze určit nejlepší univerzální vzorkovací metodu - téměř každý algoritmus byl v některém kontextu nejlepší. V konkrétní implementaci bude třeba vybírat podle předpokládaného charakteru obrazové funkce f a dalších kritérií, zejména dimenze použitého prostoru. Přestože nebyly algoritmy v prostorech vyšších dimenzí testovány, ukázal se jasný trend: pro vyšší dimenze jsou výhodnější vzorkování s náhodným přístupem a neúplným pokrytím daného intervalu.

V budoucnu by mohlo být zajímavé zkoumat další vlastnosti vzorkovacích metod, především souvislost chyby vzorkování s diskrepancí vzorků ([Zeremba68], [Shirley91]) nebo spektrální charakteristikou obrazu.

6 Literatura

[Cook84] Cook, R. L.,Porter, T., Carpenter, L.: Distributed Ray Tracing, Computer Graphics, 18, 4 (July 1984), 165-174.

[Glassner89] ed. Glassner, A. S.: An Introduction to Ray Tracing, Academic Press, London 1989.

[Měch93] Měch, R.: Barva v zobrazovacích algoritmech, diplomová práce, MFF UK Praha, duben 1993.

[Mitchell91] Mitchell, D. P.: Spectrally Optimal Sampling for Distribution Ray Tracing, Computer Graphics, 25, 4 (July 1991), 157-162.

[Shirley90] Shirley, P. S.: Physically Based Lighting Calculations for Computer Graphics, PhD thesis, University of Illinois at Urbana-Champaign, November 1990.

[Shirley91] Shirley, P. S.: Discrepancy as a Quality Measure for Sample Distributions, Eurographics '91 Conference Proccedings, 183-194.

[Whitted80] Whitted, T.: An improved illumination model for shaded display, Communications of the ACM, 23, 6 (June 1980), 343-349.

[Zeremba68] Zeremba, S. K.: The mathematical basis of monte carlo and quasi-monte carlo methods, SIAM Review, 10, 3 (July 1968), 303-314.