Realistyczna synteza animacji za pomocą map fotonów z filtrowaniem przestrzennym i czasowym

Roman Parys

praca magisterska

promotor: dr Andrzej Łukaszewski

Instytut Informatyki
Uniwersytet Wrocławski

Wrocław, 2005

Streszczenie

Głównym celem tej pracy było zaimplementowanie systemu do fotorealistycznej wizualizacji animacji modelowanych w 3D Studio. Metody wizualizacji w tym systemie opierają się na śledzeniu promieni, metodach Monte Carlo, oraz na mapach fotonów. Niniejszy dokument jest opisem metod używanych w tym programie, oraz przedstawia wyniki eksperymentów z użyciem filtrowania przestrzennego i czasowego w metodzie map fotonów. Wszystkie renderingi przedstawione w tym dokumencie są wyrenderowane za pomocą opisywanego programu. Opisane są także struktury danych i algorytmy przyśpieszające syntezę animacji. Dołączona płyta CD zawiera kod źródłowy opisywanego systemu, animacje i obrazy wykonane za pomocą tego programu oraz wersje elektroniczną tego dokumentu.

- Rozdział 1 opisuje metodę śledzenia promieni oraz przedstawia efekty, jakie można uzyskać za pomocą tej metody.
- Rozdział 2 opisuje metody Monte Carlo oraz ich zastosowanie w fotorealistycznej syntezie obrazów.
- Rozdział 3 opisuje metodę map fotonów oraz eksperymenty z użyciem filtrowania przestrzennego i czasowego map fotonów.
- Rozdział 4 opisuje sposób użycia i konfiguracji systemu do syntezy animacji.
- Rozdział 5 przedstawia wyniki szybkości syntezy obrazów za pomocą opisywanego systemu.

Spis treści

Rozdział 1

Śledzenie promieni

1.1 Wstęp

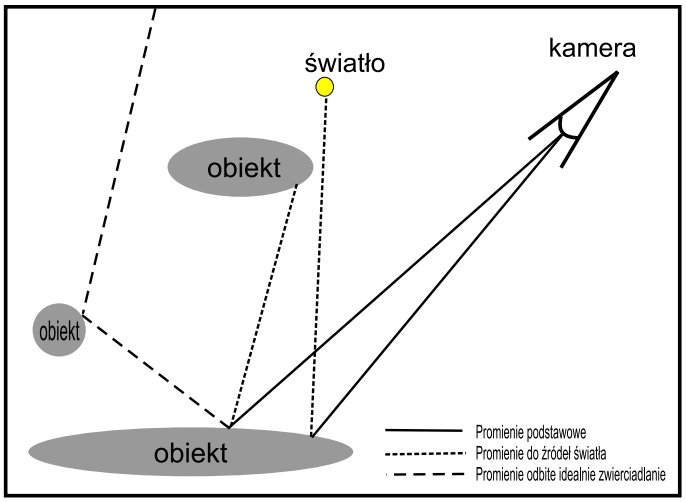
Śledzenie promieni (ang. ray tracing) jest metodą generowania obrazów i animacji na podstawie trójwymiarowych scen. Najprostsza wersja algorytmu oblicza każdy piksel obrazu wysyłając promień od oka do sceny, aby znaleźć najbliższy widoczny obiekt. Następnie obliczane jest wychodzące światło w stronę oka zgodnie z kierunkiem tego promienia przy użyciu jednego z modeli odbicia. W obliczaniu światła przychodzącego brane jest pod uwagę bezpośrednie oświetlenie punktu obiektu przez źródła światła oraz przez światło przychodzące z kierunku odbicia idealnie zwierciadlanego. Rozróżnia się następujące typy promieni:

- *Promienie podstawowe (ang. primary rays)* wysyłane z oka (kamery) w stronę sceny.
- *Promienie do źródeł swiatła (ang. shadow rays)* służą do wykrywania obszarów położonych w cieniu. Prowadzi się je od danego punktu $\boldsymbol{P}$ przecięcia promienia podstawowego lub odbitego ze sceną do źródeł światła. Jeżeli promień natrafi na jakiś obiekt po drodze do źródła światła to punkt $\boldsymbol{P}$ nie jest bezpośrenio oświetlony przez to światło. W takim przypadku nie bierzemy tego światła pod uwagę przy wyliczaniu bezpośredniego oświetlenia.
- *Promienie odbite (ang. reflected rays)* w prostej wersji algorytmu używane są do obliczania odbić idealnie zwierciadlanych.

Rysunek 1.1 pokazuje różne typy promieni. Nie ma dużych ograniczeń na typ obiektów w metodzie śledzenia promieni – obiekty mogą być np. trójkątami, wielokątami, powierzchniami parametrycznymi, lub obiektami fraktalami.

1.2 Opis sceny

Opisywany system pozwala na rendering scen dynamicznych. Obiekty mogą się poruszać, obracać i ulegać skalowaniu. Scena wczytywana jest z plików w formacie 3D Studio (*.3ds) i zawiera następujące elementy:



Rysunek 1.1: Typy promieni

- **Obiekty** – siatki złożone z trójkątów. Za pomocą trójkątów można aproksymować dowolne kształty oraz można szybko wykrywać przecięcia promienia z trójkątem. Każdy obiekt posiada swoją macierz przekształcenia M o rozmiarze 4x4, która przekształca obiekt z lokalnego układu współrzędnych do układu globalnego. Macierz M pozwala na transformacje będące złożeniami translacji, obrotów oraz skalowania. Obiekt posiada także macierz M^{-1} do transformacji promieni z globalnego układu współrzędnych do lokalnego układu współrzędnych obiektu. Gdy obiekt jest złożony z bardzo dużej ilości trójkątów, wykonujemy transformację promienia za pomocą macierzy M^{-1}, wykrywamy przecięcie i dokonujemy transformacji odwrotnej punktu przecięcia i wektora normalnego za pomocą macierzy M. Nie musimy dokonywać transformacji wszystkich wierzchołków obiektu.

- **Źródła światła** – punktowe oraz płaszczyznowe. Światła punktowe powodują powstawanie cieni o ostrych krawędziach. Światła płaszczyznowe dają miękkie cienie, jednak obliczanie wpływu takiego światła wymaga większego nakładu obliczeniowego. Format 3ds nie zawiera możliwości bezpośredniego definiowania świateł płaszczyznowych. System wczytuje opcjonalny plik rozszerzający format 3ds z definicjami świateł płaszczyznowych. W tym pliku podaje się nazwę materiału, którym pokryty jest obiekt oraz moc na jednostkę powierzchni światła. Każdy trójkąt obiektu pokrytego tym materiałem staje się płaszczyznowym źródłem światła. Macierz M obiektu zamienionego na źródło światła jest używana podczas animacji. Plik umożliwia także przedefiniowanie współczynników odbicia dla materiałów.

- **Materiały** – pokrywają obiekty i posiadają takie właściwości jak współczynniki odbicia światła (współczynnik odbicia rozproszonego i zwierciadlanego), tekstura bitmapowa pokrywająca obiekt oraz procentowy współczynnik przeźroczystości.

- **Kamera** – jest reprezentowana przez punkt zaczepienia, wektor kierunku pa-

trzenia oraz wektor skierowany w górę. Kamera posiada swoją macierz przekształcenia i podlega animacji.

- **Informacje o animacji** – pozwalają na modyfikowanie macierzy przekształceń obiektów, kamery i świateł.

1.3 Szybkie wykrywanie przecięć promień–obiekt

1.3.1 Transformacje promieni

Niech M (rozmiaru 4x4) będzie macierzą przekształcenia obiektu z lokalnego układu współrzędnych do układu globalnego. Dla promienia $R = U * t + V$ w globalnym układzie współrzędnych obiektu (U to wektor kierunku, V to punkt, z którego promień wychodzi) obliczamy promień $R' = t * M^{-1} * U + M^{-1} * V$. W ten sposób transformujemy promień R w promień R' w lokalnym układzie współrzędnych obiektu. Następnie obliczamy najbliższy punkt x przecięcia promienia R' z obiektem w lokalnym układzie współrzędnych. Następnie transformujemy punkt x do globalnego układu współrzędnych za pomocą macierzy M. W ten sposób otrzymujemy punkt przecięcia w globalnym układzie współrzędnych. Transformację punktu $V = (v_0, v_1, v_2)^T$ przez macierz M rozmiaru 4x4 obliczamy ze wzoru

$$\begin{pmatrix} v'_0 \\ v'_1 \\ v'_2 \\ 1 \end{pmatrix} = \begin{pmatrix} m_{00} & m_{01} & m_{02} & m_{03} \\ m_{10} & m_{11} & m_{12} & m_{13} \\ m_{20} & m_{21} & m_{22} & m_{23} \\ 0 & 0 & 0 & 1 \end{pmatrix} * \begin{pmatrix} v_0 \\ v_1 \\ v_2 \\ 1 \end{pmatrix} \tag{1.1}$$

Transformacje wektora $U = (u_0, u_1, u_2)^T$przez macierz M rozmiaru 4x4 obliczamy ze wzoru

$$\begin{pmatrix} u'_0 \\ u'_1 \\ u'_2 \\ 0 \end{pmatrix} = \begin{pmatrix} m_{00} & m_{01} & m_{02} & m_{03} \\ m_{10} & m_{11} & m_{12} & m_{13} \\ m_{20} & m_{21} & m_{22} & m_{23} \\ 0 & 0 & 0 & 1 \end{pmatrix} * \begin{pmatrix} u_0 \\ u_1 \\ u_2 \\ 0 \end{pmatrix} \tag{1.2}$$

Zero w ostatnim czwartym komponencie wektora U powoduje, że elementy macierzy odpowiedzialne za translacje zostają wyzerowane. Dzieje się tak, ponieważ wektory wskazują kierunek, który nie zmienia się przy ich przesunięciach. Wektor normalny n' (w globalnym układzie współrzędnych) powierzchni transformowanej przez macierz M wylicza się ze wzoru

$$n' = (M^{-1})^T n \tag{1.3}$$

W powyższym równaniu $n = (n_x, n_y, n_z, 0)$ jest wektorem normalnym do płaszczyzny w punkcie x w lokalnym układzie współrzędnych.

1.3.2 Kd–drzewa

Kd–drzewa są strukturą danych dzielącą hierarchicznie przestrzeń wypełnioną elementami. Dzięki temu niektóre algorytmy wykorzystujące kd–drzewa są w stanie

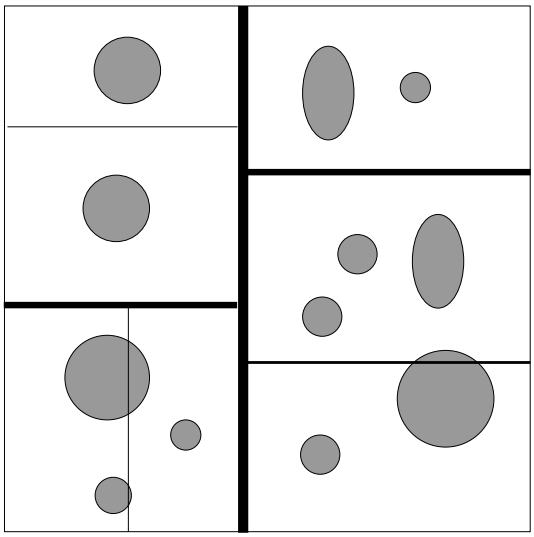
eliminować całe poddrzewa w trakcie obliczeń. Płaszczyzny podziału są prostopadłe do osi układu współrzędnych, dzięki temu obliczenie przecięcia promienia z płaszczyzną można wykonać bardzo szybko. Podziały wykonywane są kolejno za pomocą płaszczyzn P_x, P_y, P_z, gdzie P_a oznacza płaszczyznę prostopadłą do osi a. Idea hierarchicznego podziału przestrzeni przedstawiona jest na rysunku 1.2. W przedstawianym systemie kd–drzewo budujemy dla każdego obiektu i elementami są poszczególne trójkąty. Węzeł kd–drzewa zawiera płaszczyznę rozdzielającą oraz wskaźniki do poddrzew lub liści. Liść zawiera zbiór trójkątów znajdujący się w jego obszarze. Algorytm 1 buduje kd–drzewo. Metod do ustalania płaszczyzny podziału jest wiele. Najprostsza metoda polega na dzieleniu prostopadłościanu ograniczającego na dwie równe części przecinając najdłuższe krawędzie. Takie drzewa są szybko budowane, lecz nie są zawsze zbalansowane. To wydłuża późniejszy czas przy wykrywaniu przecięć. Inna metoda polega na znalezieniu takiej płaszczyzny, że ilość trójkątów po jej lewej i prawej stronie jest zbliżona. Można to rozwiązać sortując listę trójkątów według każdej płaszczyzny P_x, P_y, P_z. Miejscem podziału będzie płaszczyzna leżąca w pobliżu środkowego trójkąta. Wybieramy płaszczyznę, która przetnie jak najmniej trójkątów. Trójkąty leżące po obu stronach płaszczyzny podziału zapamiętujemy w obu poddrzewach. Takie rozwiązanie jest bardziej kosztowne przy konstrukcji, jednak produkuje bardziej zbalansowane drzewa. W opisywanym systemie zaimplementowany jest algorytm podziału prostopadłościanu na pół, kolejno według każdej płaszczyzny P_x, P_y, P_z. Algorytm 2 wykrywa przecięcia promień–trójkąt przechodząc drzewo od korzenia i odrzucając całe poddrzewa w przypadku znalezienia przecięcia po bliższej stronie płaszczyzny podziału. Czas działania przy wykrywaniu przecięć z użyciem kd–drzewa przy pewnych założeniach wynosi $O(log(n))$, gdzie n to liczba trójkątów w drzewie. W tym przypadku założeniem jest, że drzewo jest zbalansowane, oraz ilość trójkątów leżących na płaszczyźnie podziału (a więc w dwóch poddrzewach) jest bardzo niewielka. W praktyce idealne założenia jest trudno spełnić, ale mimo to kd–drzewa są bardzo efektywną strukturą danych.

Scenę reprezentuje lista obiektów (kd–drzew), każde z nich ma swój prostopadłościan ograniczający. Sprawdzenie przecięcia trójkąt–promień polega na przechodzeniu listy i sprawdzaniu czy promień przecina prostopadłościan ograniczający danego obiektu. Jeżeli następuje przecięcie z prostopadłościanem, to zostaje uruchomiony algorytm 2. Każdy promień jest transformowany do lokalnego układu współrzędnych danego obiektu.

1.4 Model odbicia światła

System używa zmodyfikowanego modelu Phonga jako modelu odbicia. Dla światła punktowego wyliczamy $L(x, \omega_i)$ – kolor punktu x widziany z kierunku ω_i z następującego wzoru (gdy dane światło jest widoczne z punktu x):

$$L(x, \omega_i) = \left(\frac{k_d}{\pi} + \frac{k_s(n+2)}{2\pi} cos(\Theta')^n \right) cos(\Theta) \frac{power}{4\pi r^2} \tag{1.4}$$



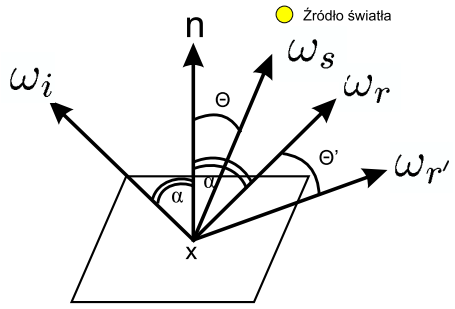
Rysunek 1.2: Hierarchiczny podział przestrzeni za pomocą kd–drzew (przypadek 2D).

Algorytm 1 Budowanie kd–drzewa

```
buduj(Węzeł &W, ZbiórTrójkątów T) {
   if (T jest mały lub osiągnięto maksymalną wysokość drzewa) {
      W.zbiór_trójkątów=T;
   else {
      po = prostopadłościan_ograniczający(T);
      W.płaszczyzna_podziału = wybierz_płaszczyznę_podziału(po);
      po_1 = prostopadłościan_ograniczający(po, T, W, przód);
      po_2 = prostopadłościan_ograniczający(po, T, W, tył);
      T1 = trójkąty_w_prostopadłoscianie(po_1, T);
      T2 = trójkąty_w_prostopadłościanie(po_2, T);
      W.W1 = nowy_wezel(po_1);
      W.W2 = nowy_wezel(po_2);
      buduj(W.W1, T1);
      buduj(W.W2, T2);
   }
```

Algorytm 2 Wykrywanie przecięcia trójkąt–promień z zastosowaniem kd–drzew.

```
// Struktura DanePrzecięcia zawiera punkt przecięcia,
// wektor normalny do płaszczyzny
// oraz identyfikator przeciętego obiektu.
boolean przecięcie(Węzeł &W, Promień R, DanePrzecięcia &D) {
   if (W jest liściem) {
      foreach (trójkąt T in W.zbiór_trójkątów)
         wykryj_przecięcie(T,R,D);
      if (było_przecięcie)
         return true;
      else
         return false;
   } else {
      if (bliższy_węzeł(W.W1, R) {
         b=przecięcie(W.W1, R, D);
         if (not b)
             b=przecięcie(W.W2, R, D);
         return b;
      } else {
         b=przecięcie(W.W2, R, D);
         if (not b)
             b=przecięcie(W.W1, R, D);
         return b;
      }
   }
}
```



$L(x, \omega_i)$, *power* (natężenie światła), k_d *i* k_s są wyrażone trzema nieujemnymi wartościami R G B dla każdego kanału (czerwonego, zielonego i niebieskiego), r to odległość między cieniowanym punktem a źródłem światła, k_d to współczynnik odbicia rozproszonego dla punktu x, k_s współczynnik odbicia zwierciadlanego dla punktu x. Współczynniki k_d i k_s spełniają nierówność:

$$k_d + k_s < (1, 1, 1)^T \tag{1.5}$$

Współczynnik n>1 powoduje powstawanie rozbłysków takich jak na rysunku 1.3. Dla małych kątów Θ' między kierunkiem odbicia $\omega_{r'}$ a kierunkiem odbicia idealnie zwierciadlanego ω_r współczynnik $cos(\Theta')^n$ jest bliski jedynce i wraz ze wzrostem kąta Θ' ten współczynnik maleje. Funkcja $cos(\Theta')^n$ ma tę własność, że im większe n tym szybciej ona maleje. Dla wielu źródeł światła sumuje się wkład każdego światła j do koloru piksela :

$$L(x, \omega_i) = \sum_{j=1}^{N} V(x, j) \left(\frac{k_d}{\pi} + \frac{k_s(n+2)}{2\pi} cos(\Theta'_j)^n \right) cos(\Theta_j) \frac{power_j}{4\pi r_j^2} \tag{1.6}$$

gdzie $V(x, j)$ jest funkcją widoczności światła j z punktu x. Gdy promień do źródła światła nie jest przesłonięty przez jakiś obiekt to funkcja przyjmuje wartość 1, w przeciwnym przypadku 0. Symbole z indeksem j odnoszą się do światła j.

Model świateł płaszczyznowych jest opisany w rozdziale 2, gdyż odwołuje się do równania oświetlenia globalnego.

1.5 Odbicia idealnie zwierciadlane

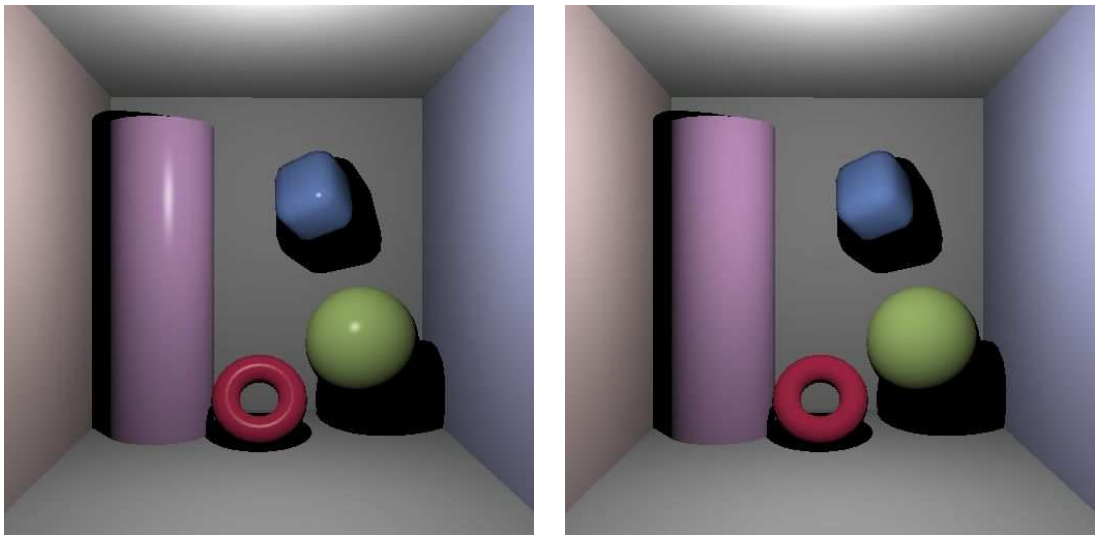
Współczynnik k_r odbicia opisuje ilość światła odbijanego w sposób idealnie zwierciadlany. We wzorze 1.6 można uwzględnić odbicia idealnie zwierciadlane formułując następujący wzór:

$$L(x, \omega_i) = \sum_{j=1}^{N} \{ V(x, j) \left(\frac{k_d}{\pi} + \frac{k_s(n+2)}{2\pi} cos(\Theta'_j)^n \right) cos(\Theta_j) \frac{power_j}{4\pi r_j^2} \} + k_r L(x, \omega_r) \tag{1.7}$$

przy założeniu:

$$(0, 0, 0)^T \leq k_r \leq (1, 1, 1)^T ,$$

gdzie ω_r jest kierunkiem odbicia idealnie zwierciadlanego dla kierunku ω_i. Dzięki temu możemy uzyskać lustra i powierzchnie odbijające w sposób idealnie zwierciadlany. Rysunek 1.4 pokazuje efekt odbić idealnie zwierciadlanych. Odbicia są



(a) (b)

Rysunek 1.3: Rozbłyski spowodowane współczynnikiem odbicia zwierciadlanego dla $k_s > 0$ [rendering (a)] oraz brak rozbłysku dla $k_s = 0$ [rendering (b)] dla $n = 64$.

realizowane za pomocą rekurencyjnego ray tracingu. Gdy promien trafi w obiekt o współczynniku $k_r > 0$ następuje odbicie promienia w kierunku idealnie zwierciadlanym i obliczenie oświetlenia padającego z tego kierunku. Rekurencyjne wywołanie się kończy gdy promień trafi w obiekt rozpraszający ($k_r = 0$) lub gdy zostanie osiągnięta maksymalna głębokość rekursji.

1.6 Przeźroczystość i załamanie promieni

Gdy promień trafi w punkt x przeroczystego obiektu oświetlenie wyliczamy ze wzoru:

$$L(x,\omega_i) = \sum_{j=1}^{N}\{V(x,j)\left(\frac{k_d}{\pi} + \frac{k_s(n+2)}{2\pi}cos(\Theta'_j)^n\right)cos(\Theta_j)\frac{power_j}{4\pi r_j^2}\}+$$

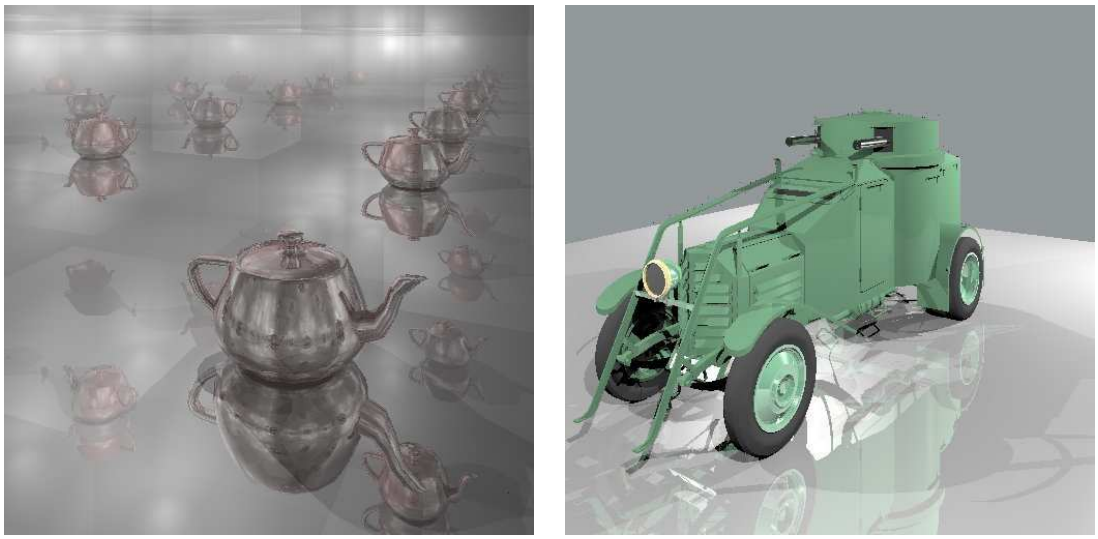
$$k_r L(x, omega_r) + k_t L(x, ref) \tag{1.8}$$

przy założeniu:

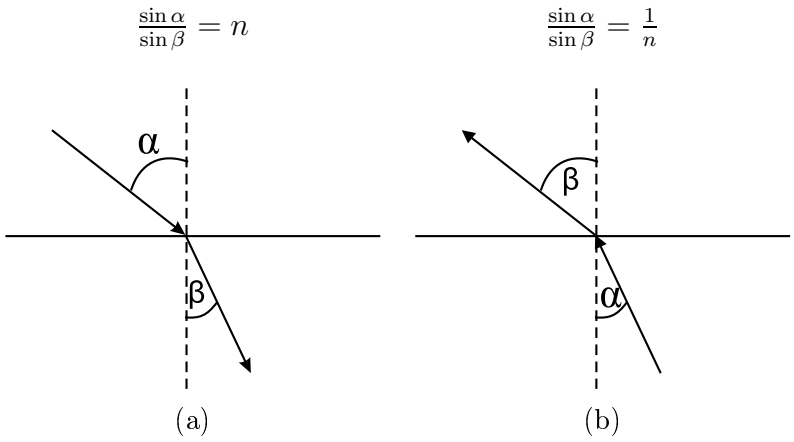
$$(0,0,0)^T \leq k_r \leq (1,1,1)^T\,, \tag{1.9}$$

$$(0,0,0)^T \leq k_t \leq (1,1,1)^T\,, \tag{1.10}$$

gdzie *ref* jest kierunkiem załamania promienia o kierunku ω_i w punkcie x, a k_t jest współczynnkiem przeźroczystości (3 wartości R, G, B), wynosi (0,0,0) dla nieprzeźroczystych obiektów i (1,1,1) dla całkowicie przeźroczystych obiektów. Prawo załamania światła ilustruje rysunek 1.5. Przykłady renderingów dla różnych współczynników znajdują się na rysunku 1.6. Tablica 1.1 podaje współczynniki załamania światła dla wybranych materiałów.



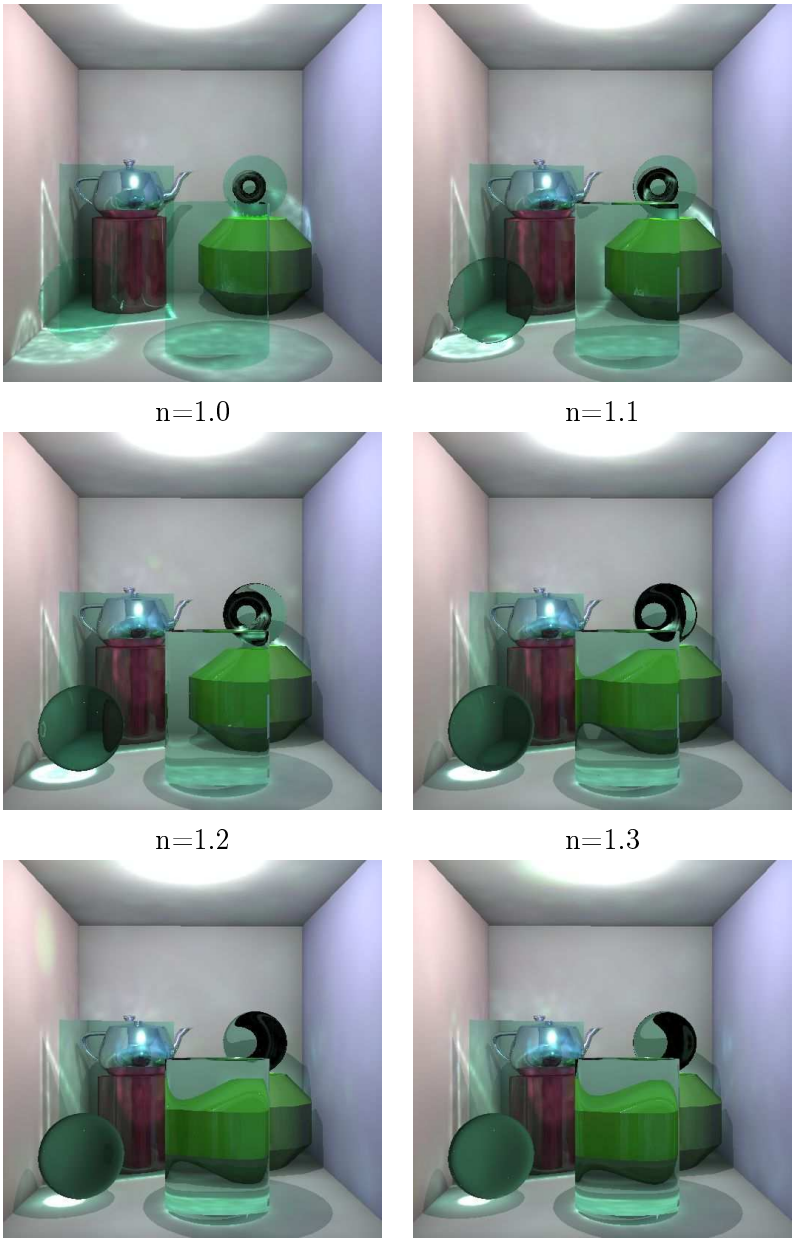
Rysunek 1.4: Renderingi przedstawiające efekt odbicia idealnie zwierciadlanego.



Rysunek 1.5: Prawo załamania promieni. Formuła (a) pokazuje związek między promieniem przechodzącym z próżni do materiału o współczynniku załamania światła n, natomiast, gdy promień wychodzi z przeroczystego obiektu w próżnie zachodzi związek (b)

1.7 Gładkie cieniowanie

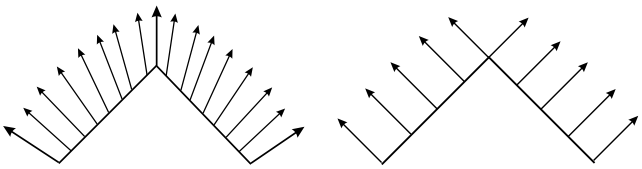
Każdy obiekt składa się z trójkątów. Aby uzyskać gładkie cieniowanie należy w każdym wierzchołku trójkąta pamiętać wektor normalny, i w każdym cieniowanym punkcie trójkąta należy używać wektora normalnego będącego kombinacją trzech wektorów normalnych z wierzchołków. Rysunek 1.7 pokazuje przykład wektorów normalnych nie interpolowanych oraz interpolowanych.



Rysunek 1.6: Porównanie różnych współczynników załamania światła n. Użyto metody mapowania fotonów aby uwidocznić skupianie światła w zależności od n. Metoda mapowania fotonów jest opisana w rozdziale 3.

Rodzaj materiału	Współczynnik refrakcji
Próżnia	1.0
Powietrze	1.0003
Woda	1.33
Alkohol	1.329
Alkohol etylowy	1.36
Aceton	1.36
Szkło (zwykłe)	1.5
Lód	1.309
Diament	2.417

Tablica 1.1: Współczynniki załamania światła dla wybranych materiałów.

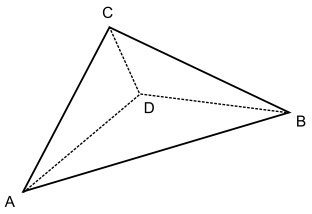


(a) (b)

Rysunek 1.7: Wektory normalne interpolowane (a) oraz nie interpolowane (b).

Funkcja wykrywania przecięcia trójkąta z promieniem **src**+**dir***t zwraca **t** (czyli punkt przecięcia) oraz współrzędne barycentryczne przecięcia. Te współrzędne spełniają równania:

$$\alpha = \frac{P_{ACD}}{P_{ABC}} \quad \beta = \frac{P_{ABD}}{P_{ABC}}$$

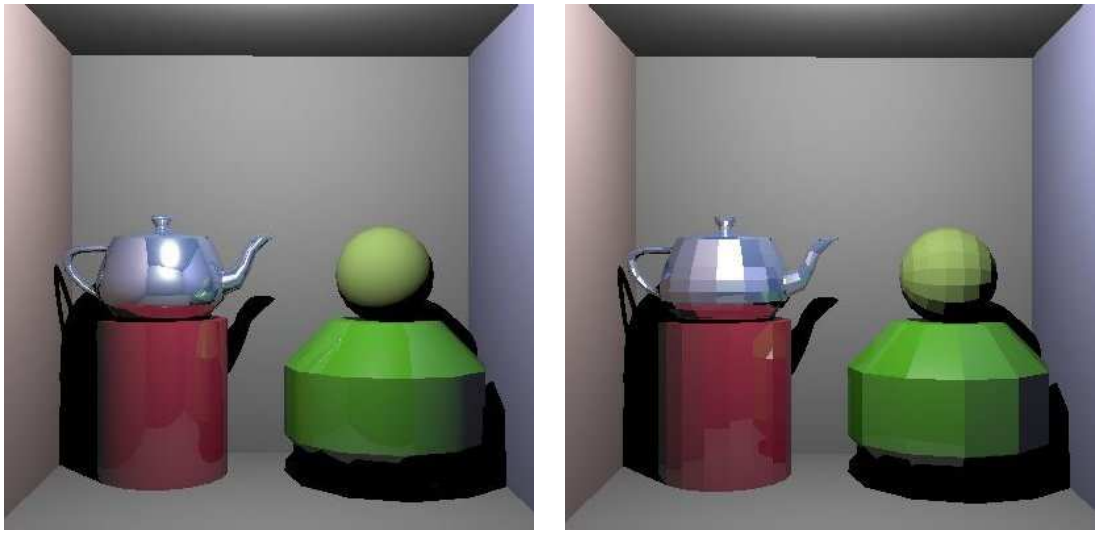


Punkt D jest punktem przecięcia promienia z trójkątem, a P_{XYZ} oznacza pole odpowiedniego trójkąta. Niech N_A , N_B , N_C oznaczają wektory normalne przy wierzchołkach A, B, C. Wtedy interpolowany wektor normalny w punkcie D wylicza się ze wzoru:

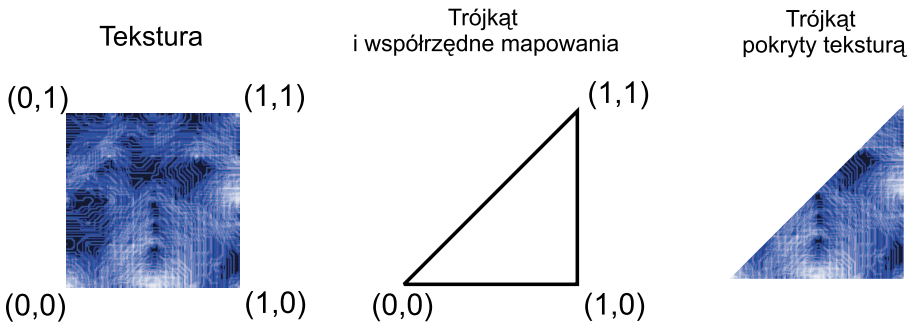
$$N_D = \beta N_A + \alpha N_B + (1 - \alpha - \beta) N_C \tag{1.11}$$

1.8 Teksturowanie

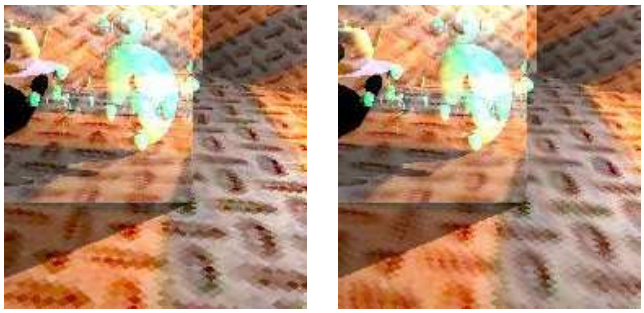
Zastosowanie współrzędnych mapowania tekstur pozwala na pokrywanie obiektów teksturami wczytywanymi z plików graficznych. Każdemu wierzchołkowi trójkąta



Rysunek 1.8: Obrazy wyrenderowane z użyciem interpolacji wektorów normalnych (a) oraz bez interpolacji (b).



Rysunek 1.9: Nakładanie tekstur na trójkąty przy pomocy współrzędnych mapowania tekstur.



(a) (b)

Rysunek 1.10: Rendering z bez interpolacji (a) i z interpolacją (b) kolorów sąsiednich pikseli.

przypisuje się wartości u i v z przedziału [0,1] odpowiadające punktowi na teksturze. Nakładanie tekstur na trójkąty prezentuje rysunek 1.9.

Funkcja wykrywania przecięcia zwraca współrzędne barycentryczne , które zostają użyte do obliczenia piksela na teksturze, którego kolor po przeskalowaniu będzie odpowiadał współczynnikowi k_d. Wartości dyskretne koloru każdego kanału z przedziału [0..255] są przekształcane w wartości rzeczywiste z przedziału [0..1]. Następnie następuje skalowanie współczynników k_d i k_s w celu spełnienia nierówności 1.5. Współrzędne piksela (x,y) na teksturze wyznacza się ze wzoru:

$$u = \alpha u_1 + \beta u_2 + (1 - \alpha - \beta)u_0$$

$$v = \alpha v_1 + \beta v_2 + (1 - \alpha - \beta)v_0$$

$$x = [u \cdot tx] \, mod \, tx$$

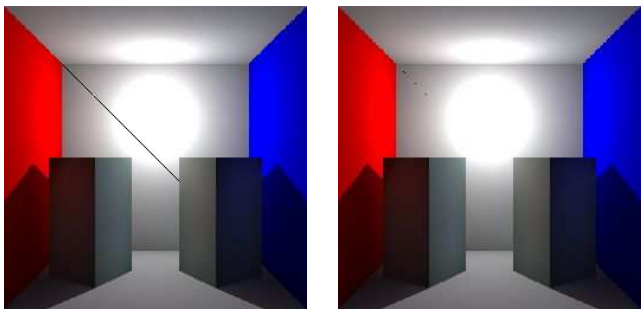
$$y = [v \cdot ty] \, mod \, ty \, ,$$

gdzie $(u_\mathrm{i}, v_\mathrm{i})$ to współrzędne mapowania tekstury dla i–tego wierzchołka trójkąta dla i=0,1,2, a tx i ty są rozmiarami tekstury. Symbol [s] oznacza zaokrąglenie liczby rzeczywistej s do najbliższej liczby całkowitej. Aby poprawić jakość teksturowania możemy interpolować liniowo kolory sąsiadujących pikseli. To podejście zmniejsza ostre przejścia między sąsiadującymi pikselami w renderingach, co jest zilustrowane na rysunku 1.10.

1.9 Błędy numeryczne

Dla kamery prostopadłej do danej powierzchni złożonej z dwóch trójkątów można zauważyć artefakty powstałe w wyniku błędów numerycznych w algorytmie Moellera. Algorytm ten znajduje przecięcie promienia z trójkątem. W zależności od typu liczb (pojedynczej lub podwójnej precyzji – float i double) artefakty są bardziej lub mniej widoczne. Rysunek 1.12 ilustruje to zjawisko. Użycie liczb typu double znacznie zmniejsza artefakty przy spadku szybkości rzędu 1,65% (tablica 1.2). W większości te artefakty są zupełnie niewidoczne, gdyż rysunek 1.12 przedstawia najgorszy przypadek artefaktów jaki udało się zauważyć. Problem występuje przy

Rysunek 1.11: Renderingi za pomocą ray tracingu z wykorzystaniem teksturowania.



(a) (b)

Rysunek 1.12: Obraz ilustruje błędy numeryczne powstające przy użyciu liczb rzeczywistych typu float (a) i double (b) przy wykrywaniu przecięć promień–trójkąt. W przypadku (a) widoczna jest czarna linia na białej scianie. W przypadku (b) zauważalny błąd wystąpił tylko dla czterech pikseli.

ustawieniu kamery o kierunku patrzenia prostopadłym do powierzchni złożonej z minimum dwóch trójkątów. Minimalne odchylenia kierunku patrzenia kamery od kierunku prostopadłego do powierzchni powodują, że błędy numeryczne są na tyle małe, że nie są przyczyną powstawania artefaktów.

Typ liczb rzeczywistych	Czas renderingu (sek.)	Ilość przecięć promien—trójkąt na sekunde
float	12.26	11584.11
double	12.42	11392.35

Tablica 1.2: Porównanie szybkości renderingu dla sceny cornell.3ds przy użyciu liczb typu float oraz double do wykrywania przecięć promień–trójkąt. Obraz renderowany w rozdzielczości 256x256 i z użyciem map fotonów (20000 fotonów, 350 do estymacji). Obliczenia zostały przeprowadzone na procesorze Athlon XP 2.6 (2GHz).

Rozdział 2

Metody Monte Carlo w realistycznej grafice komputerowej

2.1 Wstęp

Klasyczna metoda śledzenia promieni ma swoje ograniczenia. Nie daje ona efektów występujących w rzeczywistości, takich jak:

- **oświetlenie pośrednie** powstaje w wyniku jednego lub więcej rozproszonych odbić światła.
- **krwawienie kolorów** polega na transferze koloru między powierzchniami rozpraszającymi. Przykładem jest zbliżenie do siebie białej i czerwonej kartki. W wyniku krwawienia kolorów biała kartka będzie miała różowe zabarwienie. Efekt krwawienia kolorów zilustrowany jest na rysunku 2.1.
- **kaustyki** powstają w wyniku padania skupionych przez obiekty przeźroczyste wiązek światła na powierzchnie. Miejsca padania skupionych wiązek światła są jaśniejsze. Kaustyki są także generowane przez powierzchnie odbijające zwierciadlanie. Światło odbite od powierzchni zwierciadlanych padając na inne powierzchnie powoduje ich rozjaśnienia.
- **miękkie cienie** generowane są przez światła płaszczyznowe. Miękkie cienie są opisane w podrozdziale 2.10.

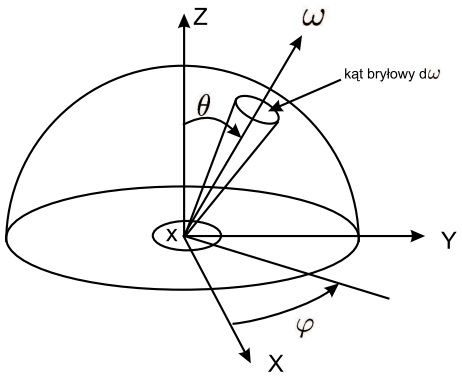
Kaustyki są realizowane dopiero przez metodę map fotonów opisaną w rozdziale 3.

2.2 Pojęcia wstępne

2.2.1 Parametryzacja sferyczna kierunków

Parametryzacja sferyczna kierunków polega na reprezentowaniu wektora kierunku ω za pomocą pary liczb (θ, φ) odpowiadającym kątom na rysunku 2.2.

Rysunek 2.1: Przykład krwawienia kolorów wzięty z rzeczywistości.



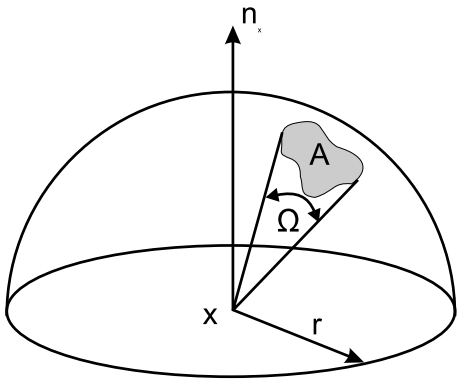
Rysunek 2.2: Parametryzacja sferyczna.

2.2.2 Kąt bryłowy

Kąt bryłowy Ω jest miarą powierzchni A na sferze o promieniu r. Wyraża się wzorem

$$\Omega = \frac{A}{r^2} \, .$$

Kątem bryłowym w jakim jest widziany obiekt z punktu x jest pole powierzchni rzutu tego obiektu na jednostkową sferę. Jednostką miary kątów bryłowych jest steradian (sr). Jeżeli $r = 1$, to kąt bryłowy jest równy powierzchni A na jednostkowej sferze.



Rysunek 2.3: Kąt bryłowy.

Równanie oświetlenia wyrażonego wzorem 2.2 w podrozdziale 2.3 będziemy całkować po półsferze, ponieważ będziemy brać pod uwagę wszystkie możliwe kierunki, z których pada światło. Miarą tego całkowania będzie różniczkowy kąt bryłowy $d\omega$ (rysunek 2.2). Jego zależność od współrzędnych sferycznych wyraża się wzorem

$$d\omega = \sin(\theta) d\theta d\varphi \ .$$

2.2.3 Radiancja i irradiancja

Radiancja jest wielkością radiometryczną wyrażoną w jednostkach $W/(sr * m^2)$ i jest oznaczana $L(x, \omega)$ dla punktu sceny x oraz kierunku ω. Radiancja jest mocą promienistą na jednostkowy kąt bryłowy na jednostkową powierzchnię. Zaletą tej jednostki jest jej niezmienność względem odległości. W praktyce oznacza to, że możemy ją łatwo obliczać za pomocą śledzenia promieni, gdyż jej wartość nie ulega zmianie w trakcie przemieszczania się przez przestrzeń (oczywiście przy założeniu, że przemieszcza się w próżni i nie napotyka po drodze obiektów). Irradiancja mierzy natężenie napromieniowania i opisuje stosunek mocy padającej na powierzchnię do pola tej powierzchni. Irradiancja wyraża się w jednostkach W/m^2.

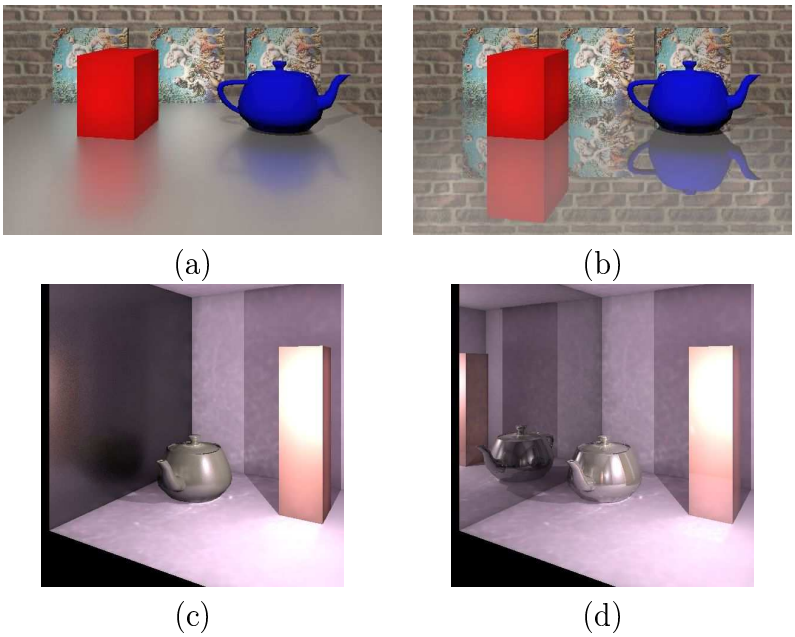
2.2.4 Funkcja BRDF

Do opisu odbicia światła będziemy używać oznaczeń z rysunku 2.5. Zakłada się, że przedstawione tam kierunki są znormalizowane. Upraszcza to liczenie cosinusów kąta między nimi do postaci prostych iloczynów skalarnych. Do opisu funkcji odbicia niezbędne są kierunki odbicia (ω_r) oraz padania (ω_i). Pozostałe kierunki, jak również wektor połówkowy ω_h między ω_r i ω_i mogą być również używane w zależności od modelu. W rzeczywistości spotykamy się z następującymi rodzajami odbicia:

- **odbicie rozproszone** – występuje wtedy, gdy światło padające na powierzchnie jest rozpraszane we wszystkich możliwych kierunkach z taką samą mocą.
- **odbicie zwierciadlane** – występuje dla gładkich powierzchni, gdy światło padające jest odbijane pod tym samym kątem względem wektora normalnego do powierzchni, i nie jest rozpraszane w żadnym innym kierunku. Mamy tu do czynienia z odbiciem idealnie zwierciadlanym. W rzeczywistości trudno jest idealne lustro, jednak wykorzystanie tego prostego modelu pozwala realizować odbicie za pomocą jednego promienia odbitego zgodnie z zasadą odbicia idealnie zwierciadlanego.
- **odbicie z połyskiem** – występuje, gdy duża część światła odbija się w stożku sąsiednich kierunków najczęściej wokół kierunku odbicia idealnie zwierciadlanego.

W praktyce odbicie jest często kombinacją wyżej wymienionych modeli. Wkład poszczególnych składowych odbicia zależy od rodzaju powierzchni.

Funkcja BRDF (bidirectional reflectance distribution function) definiuje dwukierunkową zdolność do odbijania światła przez powierzchnie. Funkcja BRDF wyraża stosunek radiancji odbitej w punkcie x w kierunku ω_r do irradiancji padającej na



Rysunek 2.4: Efekt połysku ((a) i (c), funkcja BRDF dana wzorem 2.1) oraz efekt odbicia idealnie zwierciadlanego ((b) i (d)).

ten punkt z kierunku ω_i. Funkcja BRDF jest oznaczona $f_r(x, \omega_i, \omega_r)$ i opisuje ilość światła, które jest odbijane w kierunku ω_r w zależności od kierunku padania ω_i. Założeniem jest, że odbijalność jest niezależna od obrotów powierzchni wokół wektora normalnego. Zasada Helmholtza polega na symetryczności funkcji BRDF względem kierunku padania i odbicia, czyli zachodzi równość $f_r(x, \omega_i, \omega_r) = f_r(x, \omega_r, \omega_i)$. Implementowany system zawiera funkcję

$$f_r(x, \omega_i, \omega_r) = \frac{k_d}{\pi} + \frac{k_s(n+2)}{2\pi} \cos^n(\omega_s, \omega_r) \ , \tag{2.1}$$

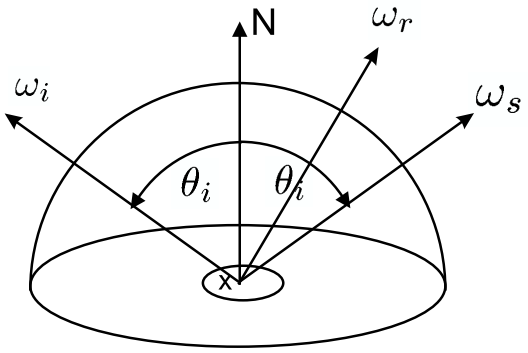
przy warunku $k_d + k_s < 1$ dla $k_d, k_s \in [0, 1)$. Funkcja 2.1 nazywana jest poprawionym modelem Phonga, gdyż przeskalowanie k_d i k_s pozwala na zachowanie energii podczas rozwiązywania równania oświetlenia opisanego w podrozdziale 2.3. Wyrażenie $\cos^n(\omega_s, \omega_r)$ w równaniu 2.1 odpowiada za połysk materiału, gdyż funkcja posiada największą wartość w kierunku odbicia zwierciadlanego, i jest malejąca zgodnie z funkcją $\cos^n(t)$ dla zwiększającego się kąta t między kierunkiem odbicia idealnie zwierciadlanego a kierunkiem odbicia.

2.3 Równanie oświetlenia

Równanie oświetlenia (nazywane równaniem renderingu) wyraża się wzorem

$$L_r(x, \omega_r) = L_e(x, \omega_r) + \int_\Omega f_r(x, \omega_i, \omega_r) L_i(x, \omega_i) \cos(\theta_i) d\omega_i \ . \tag{2.2}$$

Idea równania 2.2 jest taka, że radiancja odbita w punkcie x w kierunku ω_r jest sumą radiancji emitowanej z tego punktu oraz radiancji odbitej w tym punkcie w



Rysunek 2.5: Geometria odbicia.

kierunku ω_r. Radiancja przychodząca do punktu x przychodzi ze wszystkich możliwych kierunków. Z tego powodu całkowanie odbywa się po półsferze kierunków Ω. Jak widać jest to równanie rekurencyjne o nieskończonej głębokości rekursji w domenie ciągłej. Z tego powodu rozwiązanie analityczne tego równania jest niemożliwe dla ogólnego przypadku.

2.4 Metody Monte Carlo

Metody Monte Carlo służą do przybliżonego obliczania całek funkcji za pomocą losowych próbek, będących wartościami funkcji w losowych punktach. Użyjemy prostego przykładu, aby pokazać ideę tych metod. Mamy funkcję $f(x)$ zadaną na przedziale $[a, b]$. Całkę

$$I = \int_a^b f(x)dx \tag{2.3}$$

przybliżamy przez sumę

$$I' = (b-a)\frac{1}{N}\sum_{i=1}^{N} f(x_i) \ , \tag{2.4}$$

gdzie x_i to losowe punkty z przedziału $[a, b]$ losowane z jednorodnego rozkładu prawdopodobieństwa. Czasem zdarza się, że funkcja ma bardzo dużą wartość na wąskim podprzedziale, a jest bliska zeru poza tym podprzedziałem. W tym przypadku rozwiązanie przybliżone będzie niedokładne, gdyż większość próbek wpadnie w przedział o wartościach funkcji bliskich zeru. Gdy mamy jakąś informacje o funkcji możemy próbować zwiększać prawdopodobieństwo losowania wartości mających większy wpływ na rozwiązanie metodą opisaną w następnym podrozdziale.

2.5 Próbkowanie ważone

Gdy wartości losujemy z rozkładem gęstości opisanym funkcją $p(x)$ używamy wzoru

$$I' = \frac{1}{N}\sum_{i=1}^{N} \frac{f(x_i)}{p(x_i)} \ . \tag{2.5}$$

Dla próbkowania jednorodnego we wzorze 2.5 używaliśmy funkcji $p(x) = (b-a)^{-1}$. Odpowiednie dobranie $p(x)$ pozwala na zmniejszanie wariancji rozwiązania.

2.6 Konstruowanie metod Monte Carlo dla potrzeb realistycznej grafiki komputerowej

Aby rozwiązać równanie 2.2 przybliżamy je skończoną ilością losowych próbek zgodnie ze wzorem:

$$L_r(x,\omega_r) \approx L_e(x,\omega_r) + \frac{1}{N}\sum_{i=1}^{N}\frac{f_r(x,\omega_i,\omega_r)L_i(x,\omega_i)\cos(\theta_i)}{p(x_i)} \,. \tag{2.6}$$

Wzór na przybliżanie wartości funkcji za pomocą losowych próbek nazywamy estymatorem. Jak widać rekurencja we wzorze 2.6 jest nadal nieskończona. Metodą ograniczania rekursji jest ustalenie stałej głębokości rekursji. Takie rozwiązanie zwiększa bias. Bias to jeden z błędów estymatora. Estymator może popełnić dwa rodzaje błędów. Pierwszym z nich to szum, który może być wyeliminowany przez nieskończoną liczbę próbek. Drugi błąd to bias i nie moze on być wyeliminowany poprzez zwiększenie liczby próbek do nieskończoności. Taki estymator jest nazywany obciążonym (biased estimator). Aby zmniejszyć szum zwiększa się ilość próbek, oraz stosuje się funkcje dobrze dobrane funkcje gęstości prawdopodobieństwa. Aby wyeliminować bias powstający w wyniku skończonej rekursji stosuje się Rosyjską Ruletkę. Metoda jest opisana w podrozdziale 2.8.

2.7 Stochastyczne śledzenie promieni

Metoda polega na znalezieniu najbliższego punktu x przecięcia promienia pierwotnego ze sceną. Aby policzyć radiancję odbitą L_r generuje się N promieni (próbek) w losowych kierunkach. Dla każdego z nich obliczamy przecięcie i wywołujemy metodę rekurencyjnie aby policzyć L_i dla każdej próbki i, To podejście powoduje bardzo szybkie rozrastanie się ilości wygenerowanych promieni (wykładniczo od głębokości rekursji). Dodatkowym problemem jest to, że promienie wygenerowane na najgłębszym poziomie rekursji mają niewielki wpływ na rozwiązanie.

2.8 Śledzenie ścieżek

Śledzenie ścieżek zakłada branie tylko jednej próbki ($N = 1$ we wzorze 2.6). Dzięki temu drzewa promieni nie rozrastają się w wykładniczo. Ponieważ wariancja takiego rozwiązania będzie duża, należy dla każdego piksela na ekranie wywołać procedurę śledzenia ścieżki 500–1000 razy i uśrednić wynik. Taka ilość ścieżek na jeden piksel gwarantuje dobrą dokładność rozwiązania. Optymalizacja tej meody polega na generowaniu oprócz jednego losowego promienia (próbki) dodatkowego promienia do losowego źródła światła i dodania wkładu oświetlenia bezpośredniego do rozwiązania.

Algorytm 3 Śledzenie ścieżek z uwzględnieniem oświetlenia bezpośredniego.

```
Kolor Radiancja(Promień r) {
   if (r trafia w punkt x sceny) {
      k   = wylosuj_światło();
      p_k = prawdopodobieństwo wylosowania światła k;

      // oświetlenie bezpośrednie policzone za pomocą jednego promienia
      L_d = oświetlenie bezpośrednie punktu x przez światło k;
      r1  = wygeneruj losowy promień nie trafiający w źródło światła;
      return L_e(x)+k_d*( L_d/p_k + Radiancja(r1) );
   } else {
      return KolorTła.
   }
}
```

Założenie, że odbicia w scenie są rozproszone (powierzchnie lambertowskie) pozwala na eleganckie sformułowanie algorytmu śledzenia ścieżek. Dla powierzchni lambertowskich funkcja BRDF przyjmuje prostą postać $f_r(x, \omega_i, \omega_r) = \frac{k_d}{\pi}$, gdzie k_d to współczynnik odbicia rozproszonego. Dzięki temu wzór na estymacje za pomocą jednej próbki przyjmuje postać

$$L(x, \omega_r) \approx L_e(x, \omega_r) + \frac{k_d L_i(x, \omega_i) cos(\theta_i)}{\pi p(x_i)} \, . \tag{2.7}$$

Dodatkowa optymalizacja polega na dobraniu gęstości prawdopodobieństwa $p(x_i)$ w ten sposób, aby prawdopodobieństwo wylosowania kierunku równoległego do płaszczyzny było zerowe. Dobra funkcja gęstości w tym przypadku to $p(\theta_i) = \frac{\cos(\theta_i)}{\pi}$. Upraszcza ona wzór 2.7 do wzoru

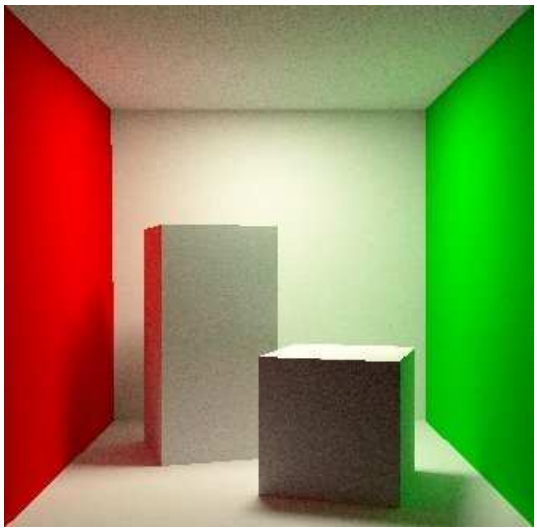
$$L(x, \omega_r) \approx L_e(x, \omega_r) + k_d L_i(x, \omega_i) \, .$$

Do tego równania można dołożyć wkład bezpośredniego oświetlenia, bo prawdopodobieństwo wylosowania kierunku do źródła światła jest małe. W tym celu można wylosować jedno światło z pewnym prawdpodobieństwem i zmodyfikować jego moc ze wzoru (p_i to prawdopodobieństwo wylosowania światła i)

$$P^*_{light_i} = \frac{P_{light_i}}{p_i} \, .$$

Dzielenie przez p_i rekompensuje fakt, że w danym punkcie ścieżki nie bierzemy wkładu wszystkich świateł do rozwiązania. Na jeden piksel wykonujemy wiele razy algorytm śledzenia ścieżki, więc każde światło ma szanse zostać uwzględnione w obliczeniu. Metoda losowania jednego źródła światła zmniejsza ilość promieni w każdym kroku algorytmu.

Rosyjska ruletka służy do przerywania ścieżek. Używamy zmiennej losowej $\xi \in [0, 1]$ o jednostajnym rozkładzie prawdopodobieństwa aby podejmować decyzje o zaprzestaniu dalszego śledzenia ścieżki lub o rodzaju odbicia danego promienia (k_d



Rysunek 2.6: Obraz wygenerowany metodą śledzenia scieżek ze światłem płaszczyznowym.

i k_s są współczynnikami odbicia spełniającymi warunek $k_d + k_s < 1$ i $k_d, k_s \in [0, 1)$):

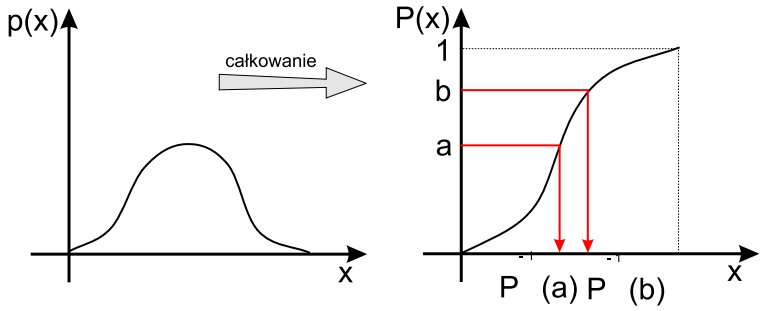
$$\begin{array}{lcl} \xi \in [0, k_d] & \longmapsto & \text{odbicie rozproszone} \\ \xi \in [d, k_d + k_s] & \longmapsto & \text{odbicie zwierciadlane} \\ \xi \in [k_d + k_s, 1] & \longmapsto & \text{terminacja ścieżki} \end{array}$$

2.9 Próbkowanie zgodne z BRDF

Metody Monte Carlo wymagają pobierania losowych próbek, aby przybliżać wartość równania oświetlenia. Metody Monte Carlo gwarantują zbieżność do prawidłowego rozwiązania przy zwiększaniu ilości próbek do nieskończoności. W praktyce okazuje się, że próbkowanie jest kosztowną operacją – wymaga liczenia przecięcia promienia ze sceną. Zbyt mała ilość próbek powoduje szum na generowanych obrazach. Zwiększanie ilości próbek zmniejsza szum, jednak wydłuża czas renderingu. Dobrym rozwiązaniem jest próbkowanie ważone o funkcji gęstości prawdopodobieństwa oparte na funkcji BRDF. Dzięki temu większe prawdopodobieństwo mają kierunki, z których przychodzi więcej światła. Takie podejście powoduje zmniejszenie szumu poprzez redukcje wariancji w rozwiązaniu równania oświetlenia.

2.9.1 Generowanie liczb losowych o niejednorodnych gęstościach

W przypadku, gdy chcemy generować liczby losowe α_i zgodnie z gęstością zadaną przez funkcje $p(x)$ zdefiniowanej na przedziale $[x_{min}, x_{max}]$ używamy liczb ϵ_i o równomiernym rozkładzie prawdopodobieństwa z przedziału $[0, 1]$ i dokonujemy ich trans-



Rysunek 2.7: Generowanie liczb losowych o dowolnych gęstościach.

formacji. Dystrybuanta $P(x)$ dana jest wzorem

$$P(x) = Prob(\alpha < x) = \int_{x_{min}}^{x} p(z)dz$$

W celu otrzymania α_i dokonujemy transformacji ϵ_i:

$$\alpha_i = P^{-1}(\epsilon_i) \ ,$$

gdzie P^{-1} jest funkcją odwrotną do P. Gdy funkcji P nie da się odwrócić, należy użyć metod numerycznych.

2.9.2 Generowanie dwuwymiarowych liczb losowych o niejednorodnych gęstościach

Gdy mamy zmienną losową $\alpha = (\alpha_x, \alpha_y)$ z dwuwymiarową funkcją gęstości prawdopodobieństwa zdefiniowaną na $[x_{min}, x_{max}] \times [y_{min}, y_{max}]$, potrzebujemy dwuwymiarowej dystrybuanty:

$$F(x,y) = Prob(\alpha_x < x \; oraz \; \alpha_y < y) = \int_{y_{min}}^{y} \int_{x_{min}}^{x} f(x,y)dxdy$$

Najpierw wybieramy x_i zgodnie z rozkładem brzegowym $F(x, y_{max})$ a następnie wybieramy y_i zgodnie z $F(x_i, y)/F(x_i, y_{max})$. Gdy $f(x, y)$ może być wyrażona za pomocą $g(x) * h(y)$ możemy użyć technik jednowymiarowych dla g i h.

2.9.3 Szybkie generowanie losowych kierunków zgodnie ze zmodyfikowanym modelem Phonga

Aby efektywnie wybierać kierunki próbkowania, użyjemy funkcji BRDF zdefiniowanej wzorem 2.1 dla $n = 64$. Funkcja gęstości prawdopodobieństwa będzie wyrażona wzorem

$$p(\theta) = \frac{\cos^{64}(\theta)}{\int_0^{\pi/2} \cos^{64}(\theta')d\theta'} \tag{2.8}$$

Z tą gęstością będziemy losować odchylenia od wektora odbicia zwierciadlanego. Składowe θ i φ kierunku ω w reprezentacji sferycznej są losowane oddzielnie, ponieważ

φ reprezentuje jedynie obrót wokół wektora odbicia zwierciadlanego i nie występuje w funkcji BRDF (będzie losowany z równomiernym rozkładem prawdopodobieństwa). Nie jest możliwe analityczne odwrócenie dystrybuanty, gdyż wyraża ona się wzorem

$$P(x) = \int_o^x p(\theta)d\theta = \alpha_0 x + \sum_{i=1}^{32} \alpha_i \sin(x)\cos^{2i-1}(x)$$

dla pewnych współczynników α_i. Funkcja została odwrócona numerycznie. Została utworzona tablica $A = (P(t_k), t_k)$ (posortowana po pierwszej kolumnie), taka że $A[\lfloor \epsilon_i \cdot 1000 \rfloor] = \theta_i$, gdzie ϵ_i jest liczbą losową z przedziału $[0, 1]$ (rozkład równomierny) a θ_i jest liczbą losową generowaną z rozkładu 2.8. To jest bardzo szybkie rozwiązanie, bo wymaga jedynie jednego mnożenia i rzutowania aby obliczyć odpowiedni indeks w tablicy. Aby uzyskać lepsze rezultaty należy wartość θ_i wyznaczyć poprzez interpolowanie sąsiednich indeksów tablicy. Wtedy kierunek $\omega = (\theta, \varphi)$ należy obliczyć ze wzoru

$$\theta = ((1 - \{\epsilon_i \cdot 1000\}) * A[\lfloor \epsilon_i \cdot 1000 \rfloor] + (\{\epsilon_i \cdot 1000\}) * A[\lfloor \epsilon_i \cdot 1000 \rfloor + 1])$$
$$\varphi = 2\pi\epsilon_i^{'} ,$$

gdzie $\{x\}$ oznacza część ułamkową z x, a $\lfloor x \rfloor$ oznacza część całkowitą x. Kierunek $\omega_r = (0, 0)$.

2.10 Miękkie cienie

Płaszczyznowe źródła światła powodują, że granice cieni obiektów są rozmyte. Ilustruje to rysunek 2.6. Punktowe źródła światła dają ostre granice cieni. Z równania renderingu (wzór 2.2) wyprowadzimy wzór na obliczanie wkładu płaszczyznowych źródeł światła. Zamiast całkować po wszystkich kierunkach, całkowanie będzie przeprowadzane po kierunkach w stronę źródła światła, a dokładniej po jego powierzchni. Korzystamy ze związku

$$d\omega = \frac{dA\cos(\theta')}{||x' - x||^2} \tag{2.9}$$

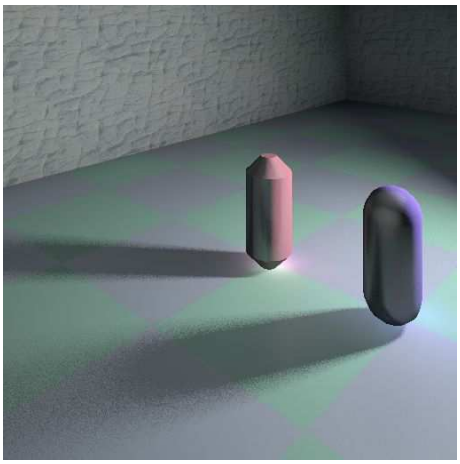
gdzie x' jest punktem na źródle światła, a θ' to kąt między wektorem normalnym do powierzchni światła w punkcie x' a wektorem $x - x'$. Wstawiając zależność 2.9 do wzoru 2.2 otrzymujemy

$$L_r(x, \omega_r) = L_e(x, \omega_r) + \int_A f_r(x, \omega_i, \omega_r)V(x, x')L_e(x', \omega_i)\frac{\cos(\theta')\cos(\theta)}{||x' - x||^2}dA , \tag{2.10}$$

gdzie $V(x, x')$ to funkcja widoczności między x i x'. Funkcja ta przyjmuje wartość 0 gdy jakiś obiekt jest między tymi punktami, 1 w przeciwnym przypadku. Estymator równania 2.10 dla $p(x) = \frac{1}{A}$ (próbkowanie jednorodne po powierzchni światła) przybiera postać:

$$L_r(x, \omega_r) \approx \frac{A}{M}\sum_{j=1}^{M} f_r(x, \omega_i, \omega_r)V(x, x'_j)L_e(x'_j, \omega_i)\frac{\cos(\theta'_j)\cos(\theta_i)}{||x'_j - x||^2} .$$

W ten sposób obliczamy radiancję odbitą w punkcie x w kierunku ω_r na podstawie radiancji docierającej bezpośrednio od źródła światła. Dla wielu źródeł światła sumujemy ich wkład.



(a)

(b)

Rysunek 2.8: Miękkie cienie. Obraz (a) wyrenderowany z użyciem 4 próbek dla źródła światła, obraz (b) z użyciem 28 próbek.

Rozdział 3

Metoda map fotonów

3.1 Mapy fotonów

Metoda map fotonów polega na emisji fotonów ze źródeł światła. Następnie fotony są śledzone w trakcie rozchodzenia się po scenie. Interakcje fotonów ze sceną są zapamiętywane w specjalnej strukturze danych. Taka struktura jest niezależna od położenia obserwatora, co jest dodatkowym atutem tej metody. Na podstawie informacji o fotonach dokonujemy estymacji oświetlenia. Mapy fotonów pozwalają także na generowanie kaustyk, czego nie można zrobić efektywnie za pomocą standardowej metody Monte Carlo.

3.1.1 Emisja fotonu

Ten fragment przedstawia metodę emisji fotonu z jednego lub wielu źródeł światła, oraz opisuje użycie map projekcji w celu zwiększania efektywności emisji.

3.1.1.1 Emisja z pojedyńczego źródła światła

Fotony ze źródła światła powinny być emitowane zgodnie z rozkładem jego mocy. Każdy foton (początkowo) przenosi tą samą moc. Dzięki temu zmniejszymy zjawisko marnowania pamięci oraz czasu procesora na fotony o małej mocy, które niewiele wnoszą do obliczanego oświetlenia. Fotony z punktowego źródła światła emitowane są w losowych kierunkach z jednorodnym rozkładem prawdopodobieństwa. Fotony z powierzchniowych źródeł światła emitowane są z losowych punktów powierzchni światła w losowych kierunkach ograniczonych do półsfery. Kierunki są losowane z rozkładu

$$p(\theta) = \frac{cos(\theta)}{\pi},$$

gdzie θ jest kątem między wektorem normalnym płaszczyzny światła a kierunkiem emisji. Dzięki temu prawdopodobieństwo wyemitowania fotonu w kierunku prostopadłym do wektora normalnego jest zerowe, i bardziej prawdopodobne są kierunki bliskie wektorowi normalnemu. Jeżeli moc światła oznaczymy przez P_{light}, a przez n_e liczbę wyemitowanych fotonów, to moc każdego wyemitowanego fotonu

Algorytm 4 Emisja fotonu z pojedyńczego źródła światła.

```
emit_photons() {
   n_e=0;
   while (jest miejsce w mapie fotonów) {
      dir=losowy kierunek zgodnie z rozkładem mocy światła;
      pnt=losowy punkt światła;
      śledź foton z punktu pnt w kierunku dir;
      n_e = n_e + 1;
   }
   foreach (zapamiętany foton p)
      p.power=p.power/n_e;
}
```

wynosi

$$P_{photon} = \frac{P_{light}}{n_e} \tag{3.1}$$

Pseudokod emisji fotonu ze źródła światła przedstawiony jest w algorytmie 4.

3.1.1.2 Emisja fotonów dla wielu źródeł światła

W tym przypadku więcej fotonów powinno być emitowane z jaśniejszych źródeł światła niż z ciemniejszych. Takie podejście sprawia, że moc emitowanych fotonów jest zbliżona. Dzięki temu nie marnujemy zasobów pamięciowych i czasu procesora zajmując się dużą ilością fotonów o małych mocach. Jednak w praktyce może zdarzyć się sytuacja, że światło o mniejszej mocy z bliska oświetla obszar znajdujący się w polu widzenia kamery, a daleko znajduje się światło, którego duża część mocy emitowana jest w zupełnie inny obszar sceny. Dla tego pomysł emitowania większej liczby fotonów z mocniejszych źródeł światła nie zawsze prowadzi do optymalnego wykorzystania mapy fotonów.

3.1.2 Mapy rzutowania

W przypadku rzadkiej geometrii lub rzadkiego rozłożenia obiektów generujących kaustyki wykorzystywane są mapy rzutowania. Mapy rzutowania dla świateł punktowych są sferami otaczającymi te światła, podzielonymi na wiele małych komórek. W przypadku płaszczyznowych źródeł światła mapa rzutowania jest płaszczyzną i składa się z małych obszarów trójkątnych. Komórka może zawierać wartość 0 lub 1, początkowo wszystkie komórki zawierają wartość 0. Następnie fotony emitowane są ze źródeł światła w kierunku obiektów. Komórka mapy rzutowania przyjmuje wartość 1, gdy przechodzi przez nią jakiś bezpośrednio wyemitowany foton. Przy generowaniu kaustyk, gdy foton uderzy w powierzchnie rozpraszającą jego interakcja nie jest zapamiętywana oraz komórka na mapie nie zmienia wartości. Przy stosowaniu map rzutowania po etapie emisji fotonów oblicza się moc każdego fotonu

stosując zmodyfikowany wzór 3.1 :

$$P_{photon} = \frac{P_{light}}{n_e} \cdot \frac{K_z}{K}\,, \tag{3.2}$$

gdzie $K_z$to pole pokrywane przez komórki o wartości 1, a K to pole całkowite mapy rzutowania.

3.1.3 Śledzenie fotonów

Gdy foton zostanie wyemitowany, jest śledzony poprzez scenę. Kiedy foton uderza w obiekt jest odbijany, absorbowany lub jego trasa może ulec zakrzywieniu przy przejściu przez przeźroczysty obiekt. Odbicie jest dwojakiego rodzaju – rozproszone lub zwierciadlane. Do sterowania zachowaniem fotonów podczas interakcji z powierzchniami korzysta się z Rosyjskiej Ruletki. Najpierw rozważymy prosty monochromatyczny przypadek. Płaszczyzna odbijająca światło posiada współczynniki odbicia rozproszonego k_d, odbicia zwierciadlanego k_s oraz przeźroczystości k_t przy warunku $k_d + k_s + k_t \leq 1$. Używamy zmiennej losowej $\xi \in [0, 1]$ o jednostajnym rozkładzie prawdopodobieństwa do sterowania zachowaniem fotonu. Na podstawie tej zmiennej podejmujemy decyzje o rodzaju interakcji fotonu z powierzchnią podczas odbicia:

$$\begin{array}{lcl}
\xi \in [0, k_d] & \longmapsto & \text{odbicie rozproszone} \\
\xi \in [k_d, k_d + k_t] & \longmapsto & \text{ugięcie trasy fotonu (przeźroczyste obiekty)} \\
\xi \in [k_d + k_t, k_d + k_t + k_s] & \longmapsto & \text{odbicie zwierciadlane} \\
\xi \in [k_d + k_t + k_s, 1] & \longmapsto & \text{absorbcja}
\end{array}$$

W tym prostym przypadku nie musimy modyfikować mocy odbitego fotonu. Rozważmy przykład powierzchni odbijającej 50% przychodzącego światła. Załóżmy, że na tą powierzchnie trafia 1000 fotonów. Możemy odbić 1000 fotonów o połowie mocy, możemy też odbić 500 fotonów o pełnej mocy używając rosyjskiej ruletki. Dla trzech kanałów RGB współczynniki k_d, k_t i k_s zawierają 3 wartości. Z tego powodu obliczane są współczynniki

$$k_{d,avg} = \frac{1}{3}(k_{d,R} + k_{d,G} + k_{d,B})$$

$$k_{t,avg} = \frac{1}{3}(k_{t,R} + k_{t,G} + k_{t,B})$$

$$k_{s,avg} = \frac{1}{3}(k_{s,R} + k_{s,G} + k_{s,B})\,,$$

które stosuje się do rosyjskiej ruletki jako k_d, k_t i k_s. W tym przypadku moc fotonu mosi być zmodyfikowana względem prawdopodobieństwa przetrwania. Przez $P_{inc,X}$ oznaczamy moc fotonu przychodzącego, a przez $P_{refl,X}$ moc fotonu odbitego dla kanału $X \in \{R, G, B\}$. Dla odbicia rozproszonego moc fotonu modyfikuje się ze wzoru

$$\begin{pmatrix} P_{refl,R} \\ P_{refl,G} \\ P_{refl,B} \end{pmatrix} = \frac{1}{k_{d,avg}} \begin{pmatrix} P_{inc,R} k_{d,R} \\ P_{inc,G} k_{d,G} \\ P_{inc,B} k_{d,B} \end{pmatrix}$$

Dla odbicia zwierciadlanego wzór jest postaci

$$\begin{pmatrix} P_{refl,R} \\ P_{refl,G} \\ P_{refl,B} \end{pmatrix} = \frac{1}{k_{s,avg}} \begin{pmatrix} P_{inc,R} k_{s,R} \\ P_{inc,G} k_{s,G} \\ P_{inc,B} k_{s,B} \end{pmatrix}$$

W przypadku ugięcia trasy fotonu wzór jest postaci

$$\begin{pmatrix} P_{refl,R} \\ P_{refl,G} \\ P_{refl,B} \end{pmatrix} = \frac{1}{k_{t,avg}} \begin{pmatrix} P_{inc,R} k_{t,R} \\ P_{inc,G} k_{t,G} \\ P_{inc,B} k_{t,B} \end{pmatrix}$$

Kierunek odbicia jest losowy, a rozkład prawdopodobieństwa kierunku odbicia jest zgodny z funkcją BRDF (najbardziej prawdopodobne są kierunki w których wartość BRDF jest duża).

Dzięki rosyjskiej ruletce przechowujemy fotony o podobnej mocy. Dzięki temu nie marnujemy czasu procesora na fotony o małej mocy względnej przy estymacji radiancji z mapy fotonów. Kolejną zaletą rosyjskiej ruletki jest teoretyczna możliwość śledzenia ścieżek o dowolnej długości. Stosowanie ścieżek o ograniczonej długości prowadzi do błędu statystycznego (bias). Wadą jest zwiększenie wariancji w rozwiązaniu, ponieważ nie używamy dokładnych wartości dla odbicia przy skalowaniu mocy fotonu. Polegamy na próbkowaniu tych wartości, które zbiegają do prawidłowego rozwiązania przy użyciu wystarczającej ilości fotonów.

3.1.4 Zapamiętywanie interakcji fotonów ze sceną

Interakcje fotonów z powierzchniami są zapamiętywane jedynie dla powierzchni rozpraszających czyli takich, że $k_d > 0$. Dla powierzchni zwierciadlanych używamy promieni odbitych w sposób zwierciadlany. Dla innych przypadków poza obiektami przeźroczystymi zapamiętujemy fotony przy każdej interakcji. Foton może być zapamiętany wielokrotnie wzdłuż swojej scieżki. Podczas absorbcji fotonu jego interakcja z powierzchnią jest również zapamiętywania. Fotony przechowujemy w globalnej strukturze danych zwanej mapą fotonów.

3.1.5 Efektywna struktura danych dla map fotonów

Dla każdej interakcji zapamiętujemy miejsce interakcji, moc przychodzącego fotonu oraz kierunek, z którego on przyszedł. Struktura fotonu jest przedstawiona poniżej:

```
struct photon {
   float x,y,z;     // pozycja fotonu
   float power_r,   // moc fotonu
         power_g,
         power_b;
   char phi,theta;  // skompresowany kierunek przychodzący
   short flag;      // flaga używana w kd-drzewie
}
```

Algorytm 5 Balansowanie kd—drzewa

```
kddrzewo *balansuj(punkty) {
   Znajdź prostopadłościan ograniczający punkty
   Wyznacz wymiar dim, w którym prostopadłościan jest największy
   Znajdź medianę punktów w wymiarze dim
   s1 = wszystkie punkty pod medianą
   s2 = wszystkie punkty ponad medianą
   węzeł = mediana
   węzeł.lewy  = balansuj(s1)
   węzeł.prawy = balansuj(s2)
   return węzeł
}
```

Kierunek przychodzący jest odwzorowaniem współrzędnych sferycznych kierunku fotonu w 65536 zdyskretyzowanych kierunków. Na podstawie zbioru fotonów tworzone jest zbalansowane kd–drzewo. Struktura ta jest niezbędna do wyszukiwania fotonów bliskich danemu punktowi w krótkim czasie. Pesymistyczny czas lokalizacji fotonu w takiej strukturze danych wynosi $O(log(N))$ gdzie N to ilość fotonów w mapie. Drzewo jest budowane po wyemitowaniu wszystkich fotonów. Zbalansowane kd–drzewo w tym przypadku jest reprezentowane za pomocą struktury przypominającej stertę. Taka struktura jest tablicą, dla danego węzła p (indeksu tablicy) synami są węzły $2p$ (lewy) i $2p+1$ (prawy). Dzięki takiemu podejściu oszczędzamy pamięć w przypadku dużych drzew, ponieważ nie musimy pamiętać wskaźników do synów zadeklarowanych w sposób jawny.

Algorytm 5 buduje zbalansowane drzewo w czasie $O(n * log(n))$ gdzie n to ilość fotonów w mapie. W praktyce czas balansowania to kilka sekund, nawet dla map rozmiaru kilku milionów fotonów. Do estymacji radiancji wychodzącej z punktu x będzie wymagane znalezienie pewnej liczby najbliższych temu punktowi fotonów. Operacja wyszukiwania będzie wykonywana często, więc algorytm musi być szybki. Algorytm zaczyna pracę w korzeniu kd–drzewa i dodaje do listy fotony znajdujące się w pewnym ustalonym obszarze. Lista k najbliższych sąsiadów jest sortowana w ten sposób, aby najdalszy foton mogł zostać łatwo usunięty w przypadku, gdy lista jest pełna i znaleziony został bliższy foton. Zamiast naiwnego sortowania stosuje się kolejke priorytetową. W algorytmie można używać kwadratów odległości zamiast samej odległości - w ten sposób nie wykonujemy kosztownego pierwiastkowania. Algorytm 6 pozwala na podanie początkowego maksymalnego dystansu wyszukiwania. Dzięki temu możemy ograniczyć ilość przeglądanych węzłów kd-drzewa.

3.2 Estymacja radiancji z mapy fotonów

Odbita radiancja w punkcie x w kierunku ω wyraża się wzorem

$$L_r(x,\omega) = \int_{\Omega_x} f_r(x,\omega,\omega_i) L_i(x,\omega_i) \cos(\theta_i) d\omega_i \,, \tag{3.3}$$

Algorytm 6 Wyszukiwanie n najbliższych fotonów.

```
// Dla mapy fotonów, punktu x i maksymalnego
// dystansu d^2 funkcja zwraca stertę h  z najbliższymi fotonami.
// Wywołuje się ją z parametrem p=1 aby zainicjować wyszukiwanie
// w korzeniu kd-drzewa.
lokalizuj_fotony(p) {
   if (2*p+1 < liczba_fotonów) {
      dist = odległość do płaszczyzny rozdzielającej węzła p
      if (dist<0) {
         lokalizuj_fotony(2*p);
         if (dist^2 < d^2)
            lokalizuj_fotony(2*p+1);
      } else {
         lokalizuj_fotony(2*p+1);
         if (dist^2 < d^2)
            lokalizuj_fotony(2*p);
      }
   }
   dist^2 = kwadrat odległości fotonu p od punktu x
   if (dist^2 < d^2) {
      wstaw foton do kolejki priorytetowej h
      d^2 = kwadrat odległości od x do fotonu w korzeniu h
   }
}
```

gdzie Ω_x jest półsferą kierunków przychodzących, f_r jest funkcją BRDF w punkcie x, a L_i jest radiancją przychodzącą. Aby obliczyć tą całkę potrzebujemy informacji o przychodzącej radiancji. Mapa fotonów zapewnia informację o przychodzącej mocy, więc korzystamy z zależności między radiancją a mocą (A_i to powierzchnia na jaką pada moc):

$$L_i(x, \omega_i) = \frac{d^2\Phi_i(x, \omega_i)}{\cos(\theta_i) d\omega_i dA_i} \tag{3.4}$$

Teraz możemy przepisać wzór 3.3:

$$\begin{aligned} L_r(x, \omega) &= \int_{\Omega_x} f_r(x, \omega, \omega_i) \frac{d^2\Phi_i(x, \omega_i)}{\cos(\theta_i) d\omega_i dA_i} \cos(\theta_i) d\omega_i \\ &= \int_{\Omega_x} f_r(x, \omega, \omega_i) \frac{d^2\Phi_i(x, \omega_i)}{dA_i} \end{aligned} \tag{3.5}$$

Przychodząca moc Φ_i jest estymowana z mapy fotonów przez zlokalizowanie k fotonów leżących najbliżej punktu x. Każdy foton p ma moc $\Delta\Phi_p(x, \omega_p)$ i przy założeniu, że fotony przecinają powierzchnie w punkcie x otrzymujemy wzór na estymację radiancji:

$$L_r(x, \omega) \approx \sum_{p=1}^{k} f_r(x, \omega, \omega_p) \frac{\Delta\Phi_p(x, \omega_p)}{\Delta A} \,. \tag{3.6}$$

Procedura estymacji polega na stopniowym zwiększaniu sfery o środku w punkcie x aż w środku znajdzie się k fotonów. Równanie 3.6 zawiera element ΔA, który odnosi się do gęstości fotonów wokół punktu x. Można założyć, że powierzchnia wokół punktu x jest lokalnie płaska, więc obliczamy pole powierzchni przez rzut kuli na tą powierzchnie, i używamy tej wartości przy estymacji radiancji. Przy tych założeniach mamy wzór:

$$\Delta A = \pi r^2 \,, \tag{3.7}$$

gdzie r jest promieniem kuli, czyli odległością najdalszego fotonu od punktu x. Wtedy równanie 3.6 przyjmuje postać:

$$L_r(x, \omega) \approx \sum_{p=1}^{k} f_r(x, \omega, \omega_p) \frac{\Delta\Phi_p(x, \omega_p)}{\pi r^2} \,. \tag{3.8}$$

Ponieważ chcemy estymować moc przychodzącą na powierzchnię P musimy do estymacji użyć jedynie fotonów znajdujących się na tej powierzchni. Fotony znajdujące się na powierzchni P możemy odróżnić od innych po ich wektorach normalnych. Fotony leżące na powierzchni P mają wektory normalne zbliżone do wektora normalnego do P. Ten fakt pozwala na wyeliminowanie fotonów znajdujących się na innych powierzchniach. Estymacja dana wzorem 3.8 jest oparta na wielu założeniach i dokładność zależy od ilości fotonów n w mapie oraz ilości fotonów k w estymacji. Estymacja daje złe rezultaty w narożnikach oraz przy krawędziach. Rozmiar tych obszarów zależy od ilości fotonów w mapie i we wzorze. Im więcej fotonów w mapie i wzorze 3.8, tym lepsza dokładność estymacji. Jezeli zignorujemy błąd z powodu ograniczonej dokładności pamiętania pozycji, kierunku oraz mocy, wtedy

przy zwiększaniu ilości fotonów do nieskończoności w mapie fotonów dla $\alpha \in (0,1)$ mamy zbieżność do rozwiązania dokładnego:

$$\lim_{N\to\infty} \sum_{p=1}^{\lfloor N^\alpha \rfloor} f_r(x,\omega,\omega_p)\frac{\Delta\Phi_p(x,\omega_p)}{\pi r^2} = L_r(x,\omega)\ . \tag{3.9}$$

Ta formuła odnosi się do wszystkich punktów x na lokalnie płaskiej powierzchni, dla których BRDF nie zawiera funkcji delta Diraca czyli nie zachodzi dla idealnych luster. Równanie 3.9 oznacza, że możemy uzyskać dowolnie dokładną estymację radiancji używając odpowiedniej ilości fotonów. W praktyce n i k we wzorze 3.8 wyznacza się eksperymentalnie.

3.3 Kaustyki

W praktyce używa się co najmniej dwóch map fotonów. Jedna to mapa globalna, a druga zawiera jedynie informacje o kaustykch. Kaustyki powstają przez skupianie wiązek światła przez przeźroczyste obiekty zakrzywiające trasy fotonów lub przez obiekty skupiające światło przez odbicie w sposób zwierciadlany. Dodatkowo można używać trzeciej mapy fotonów do pamiętania informacji o efektach wolumetrycznych, czyli np. o mgle czy dymie. Oddzielenie mapy globalnej od kaustycznej jest bardzo ważne. Estymacje z mapy globalnej są robione z użyciem dużej liczby fotonów zebranych z większego obszaru. Dzięki temu fluktuacje stają się mniej widoczne. Estymacje z mapy kaustycznej wymagają mniejszego rozmycia czyli większej ilości fotonów w ograniczonych obszarach, gdyż kaustyki często mają ostre krawędzie. Rysunek 3.1 przedstawia efekt rozmycia kaustyk poprzez wzięcie za dużej liczby fotonów do estymacji, a więc poprzez zbyt duży obszar estymacji.

3.4 Połączenie map fotonów z oświetleniem bezpośrednim

W celu efektywnej syntezy obrazów nie możemy się jedynie ograniczać do mapy fotonów w celu obliczania radiancji. Oświetlenie bezpośrednie ma największy wpływ na efekt wizualny i może być liczone wprost dużo efektywniej. Równanie renderingu można zapisać w postaci

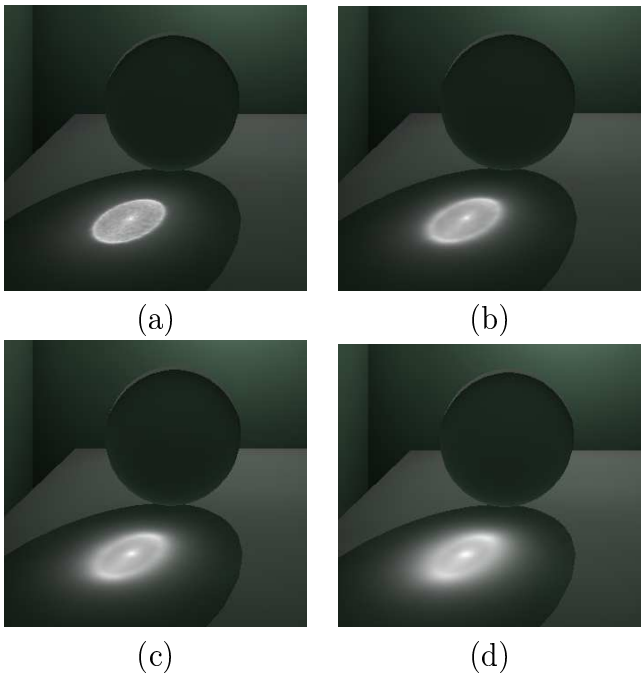
$$L_r(x,\omega_r) = L_e(x,\omega_r) + \int_\Omega f_r(x,\omega_i,\omega_r)L_i(x,\omega_i)\cos(\theta_i)d\omega_i = \tag{3.10}$$

$$\int_\Omega f_r(x,\omega_i,\omega_r)L_{i,l}(x,\omega_i)\cos(\theta_i)d\omega_i + \tag{3.11}$$

$$\int_\Omega f_{r,s}(x,\omega_i,\omega_r)(L_{i,c}(x,\omega_i) + L_{i,d}(x,\omega_i))\cos(\theta_i)d\omega_i + \tag{3.12}$$

$$\int_\Omega f_{r,d}(x,\omega_i,\omega_r)(L_{i,c}(x,\omega_i) + L_{i,d}(x,\omega_i))\cos(\theta_i)d\omega_i\ , \tag{3.13}$$

gdzie $f_r = f_{r,s} + f_{r,d}$ (BRDF jest rozbity na część rozpraszającą i zwierciadlaną) i $L_i = L_{i,l} + L_{i,c} + L_{i,d}$. Radiancja przychodząca (L_i) jest podzielona oświetlenie bezpośrednie ze świateł ($L_{i,l}$), oświetlenie pośrednie ($L_{i,d}$), oraz oświetlenie przychodzące od kaustyk ($L_{i,c}$).



Rysunek 3.1: Rozmywanie kaustyk poprzez zbyt dużą liczbe fotonów w estymacji. Mapa kaustyk zawiera 20000 fotonów, ilość fotonów w estymacji: (a) - 40, (b) - 200, (c) 500, (d) - 1000.

3.4.1 Oświetlenie bezpośrednie

Wyrażenie 3.11 reprezentuje radiancję padającą na dany punkt x bezpośrednio ze źródeł światła. Oblicza się je poprzez wysyłanie promieni do źródeł światła (shadow rays). Inne podejście łączy promienie cieni z informacją w mapie fotonów. Używa się dodatkowego rodzaju fotonów o zerowej mocy, które zapamiętywane są w mapie fotonów w obszarach cienia. Wykrywanie takich obszarów podczas śledzenia fotonów polega na przedłużeniu kierunku padania fotonu i wykrywania kolizji ze sceną promienia o tym kierunku. W punktach kolizji zapamiętujemy dodatkowe fotony. Dzięki temu w scenach, których duże obszary są nie zacienione, nie musimy używać promieni do źródeł światła, tylko użyć informacji z mapy fotonów. W miejscach, gdzie występuje cień można znaleźć fotony o zerowej mocy, i w tym przypadku dopiero używać promieni do źródeł światła. Więcej informacji w [Jensen1996]. Inne podejście używa metod offsetowych aby za pomocą jedynie jednego promienia do płaszczyznowego źródła światła stwierdzić, czy dany punkt znajduje się w całkowitym oświetleniu. Więcej informacji można znaleźć w [Łukaszewski2001].

3.4.2 Odbicie zwierciadlane

Wzór 3.12 opisuje radiancję odbitą od wysoko połyskliwych materiałów. To wyrażenie jest obliczane za pomocą standardowej metody Monte Carlo. Poprzez próbkowanie ważone opierające się na funkcji BRDF, obliczenie może być wykonane za pomocą ograniczonej liczby promieni próbkujących. W tym przypadku przy próbkowaniu używa się informacji o kaustykach z mapy kaustyk, bezpośredniej estymacji

oświetlenia pośredniego z globalnej mapy fotonów oraz oświetlenia bezpośredniego. W ten sposób unikamy rekurencyjnych wywołań funkcji generujących wiele promieni dla każdego promienia próbkującego. Oszczędność czasu renderingu jest oczywista, jakość jest także utrzymana na dobrym poziomie.

3.4.3 Obliczanie kaustyk i oświetlenia pośredniego

Wyrażenie 3.13 opisuje wpływ światła pośredniego do całkowitego rozwiązania równania renderingu. Obliczenie $L_{i,c}$ następuje poprzez bezpośrednią estymację z mapy kaustyk, natomiast $L_{i,d}$ można obliczyć na dwa sposoby. Pierwszy to bezpośrednia estymacja z mapy globalnej, drugi to metoda Final Gathering opisana w podrozdziale 3.5.

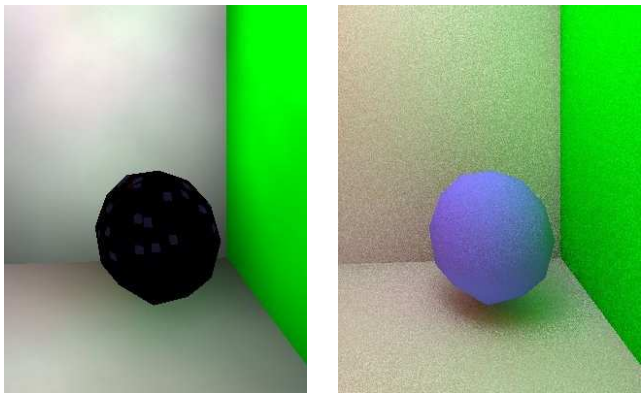
3.5 Final Gathering

Bezpośrednie estymacje radiancji z mapy fotonów powodują widoczne fluktuacje na powierzchniach sceny dla małych ilości fotonów. Ten problem jest szczególnie widoczny w animacjach. Metoda Final Gathering rozwiązuje ten problem, jednak jest kosztowna obliczeniowo dla powierzchni rozpraszających. Wymaga użycia metody Monte Carlo, jednak nie występuje tu rekurencyjne wywoływanie funkcji próbkującej, co jest zaletą tej metody. Dla powierzchni rozpraszających metoda ta wymaga 200-1000 promieni próbkujących aby zredukować szum. Stosowane podejście polega na próbkowaniu warstwowym z użyciem odpowiedniej funkcji gęstości prawdopodobieństwa. Próbkowanie warstwowe polega na podzieleniu dziedziny próbkowania na pewną ilość przedziałów i wzięcie losowej próbki z każdego przedziału. Składowa φ kierunku θ jest losowana jednorodnie z przedziału $[0, 2\pi]$ z użyciem próbkowania warstwowego. Składowe θ kierunków ω promieni próbkujących losowane są z użyciem następującej funkcji gęstości:

$$p(\theta) = \frac{\cos(\theta)}{\pi} ,$$

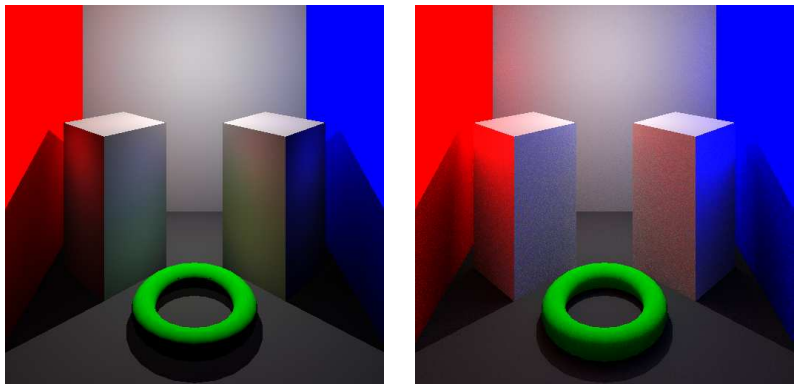
gdzie θ jest kątem między kierunkiem losowego promienia a wektorem normalnym do powierzchni. Takie podejście powoduje, że kierunki równoległe do powierzchni mają zerowe prawdopodobieństwo, natomiast największe prawdopodobieństwo mają kierunki zgodne z wektorem normalnym. Radiancja przychodząca z takich kierunków ma największy wkład w rozwiązanie ze względu na wyrażenie $\cos(\theta_i)$ w równaniu renderingu.

Metoda bezpośredniej estymacji radiancji z mapy fotonów czasami powoduje powstawanie błędów dla okrągłych obiektów. W takim przypadku pewna ilość fotonów leżących na krzywiźnie nie jest brana pod uwagę podczas estymacji ze względu na duże różnice w wektorach normalnych tych fotonów. Metoda Final Gathering rozwiązuje ten problem. Artefakty i wynik zastosowania metody Final Gathering są pokazane na rysunku 3.2. Widać tam szum spowodowany małą ilością próbek (49 próbek). Metoda Final Gathering jest wolna, bo wymaga wykrywania dużej ilości przecięć promień-trójkąt.



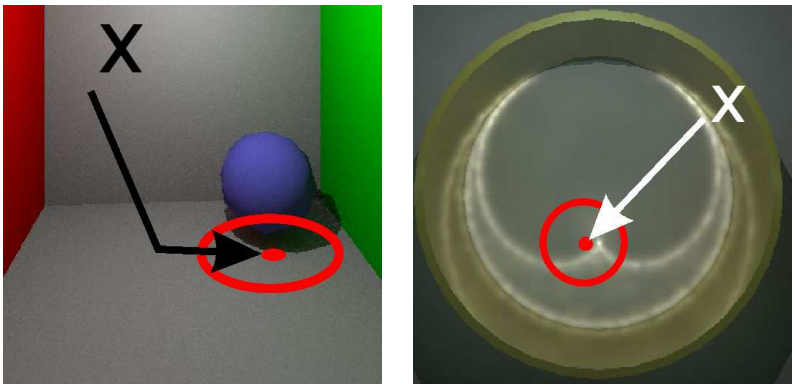
(a) (b)

Rysunek 3.2: Artefakty przy bezpośredniej estymacji radiancji (a) oraz wynik metody Final Gathering (b). Radiancja policzona jedynie na podstawie mapy fotonów. Rendering (a) trwał 8.4 sek. (rozmiar obrazu: 512x512), a rendering (b) 8 min. (rozmiar obrazu: 512x512, 49 próbek przy metodzie Final Gathering).



(a) (b)

Rysunek 3.3: Porównanie bezpośredniej estymacji radiancji z metodą Final Gathering



Rysunek 3.4: Przykłady punktów x, w których błąd estymacji spowoduje znaczne rozmycia. Okręgiem zaznaczony jest obszar zawierający fotony użyte do estymacji.

3.6 Estymacja z filtrowaniem przestrzennym

Zbyt mała liczba fotonów w mapie fotonów powoduje, że przy estymacji radiancji występują rozmazania przy krawędziach cieni, kaustyk oraz w innych miejscach, w których występuje ostra granica między obszarami o różnym natężeniu oświetlenia. Filtrowanie za pomocą jąder jest sposobem na zmniejszenie tych artefaktów oraz poprawe jakości przy tej samej liczbie fotonów.

Fotony najbliższe punktowi x są zlokalizowane w obrębie sfery, lecz do estymacji używamy tylko fotonów o podobnych wektorach normalnych (znajdujących się na tej samej płaszczynie). Fotony leżące na jednej płaszczynie są zawarte w obrębie koła o środku w punkcie x oraz promieniu r. Promień r ma długoć równą największej odległosci fotonu na tej płaszczynie od punktu x. Rozmycie powstaje, ponieważ w wyżej wymienionych przypadkach oświetlenie w danym punkcie x ma zasadniczo inne natężenie od oświetlenia w obrębie dysku wyznaczonego przez najbliższe fotony znajdujące się na tej samej płaszczynie (rysunek 3.4).

Do estymacji używamy fotonów z całego koła, również z obszarów o dużo większym lub mniejszym natężeniu oświetlenia. W przypadku prostej estymacji moc każdego fotonu w kole wpływa na wynik niezależnie od jego odległości od punktu, w którym obliczamy oświetlenie. Jądra pozwalają na nadanie tym fotonom wag, z jakimi będą uwzględniane w estymacji.

Jądrem nazywamy funkcję $K : R^2 \rightarrow R$ spełniającą równanie

$$\int_{\{(a,b)\in R^2 | a^2+b^2 \leq r^2\}} K(a,b) = 1$$

Funkcja K jest dwuargumentowa, jednak rozważane jądra będą nadawały takie same wagi dla fotonów o takiej samej odległości od punktu x. Z tego powodu parametrem funkcji K będzie jedynie odległość od punktu x. Niech d_p oznacza odległość fotonu p od punktu x, k to liczba fotonów w obrębie dysku użyta do estymacji. Wówczas wzór na estymację radiancji przyjmuje postać:

$$L(x,i) = \sum_{p=1}^{k} f_r(x,i,\psi_{i,p}) \Delta\Phi_p(x,\psi_{i,p}) K(d_p)$$

System używa następujących jąder:

- jądro proste równoważne brakowi filtrowania:

$$K(d) = \frac{1}{\pi * r^2} \tag{3.14}$$

- jądro stożkowe:

$$K(d) = \frac{3}{\pi * r^2}(1 - \frac{d}{r}) \tag{3.15}$$

- jądro gaussowskie:

$$K(d) = \frac{e^{\frac{-d^2}{2(r/2)^2}}}{2\pi(r/2)^2} \tag{3.16}$$

- jądro Epanechnikova:

$$K(d) = \frac{-2}{r^4\pi}(d + r)(d - r) \tag{3.17}$$

Użycie jądra prostego powoduje, że wzór na estymacje radiancji jest tożsamy z brakiem filtrowania. W każdym z pozostałych przypadków funkcja K maleje przy zwiększającej się odległości d. Pozwala to na nadanie mniejszych wag dla fotonów leżących w większej odległości od punktu x, zmniejszając błąd w przypadkach opisanych powyżej. Jądro gaussowskie ma objętość mniejszą niż 1 poprzez obcięcie dziedziny tej funkcji do koła o maksymalnej długości promienia r. Wartości funkcji Gaussa poza tym przedziałem są bardzo małe, a jądro daje rezultaty porównywalne z jądrem Epanechenikova i stożkowym.

Rysunki 3.5 i 3.6 przedstawiają scenę cor3.3ds. Oświetlenie na tych rysunkach wyliczone jest jedynie na podstawie map fotonów, aby przedstawić różnice w działaniu różnych jąder. Rysunek 3.7 przedstawia scenę cor3.3ds z użyciem metody polegającej na połączeniu oświetlenia bezpośredniego wyliczonego za pomocą promieni do źródeł światła oraz oświetlenia pośredniego wyliczonego na podstawie mapy fotonów. Rysunek 3.8 przedstawia scenę ring3.3ds wyrenderowaną przy użyciu różnych jąder do filtrowania kaustyk o ostrych krawędziach.

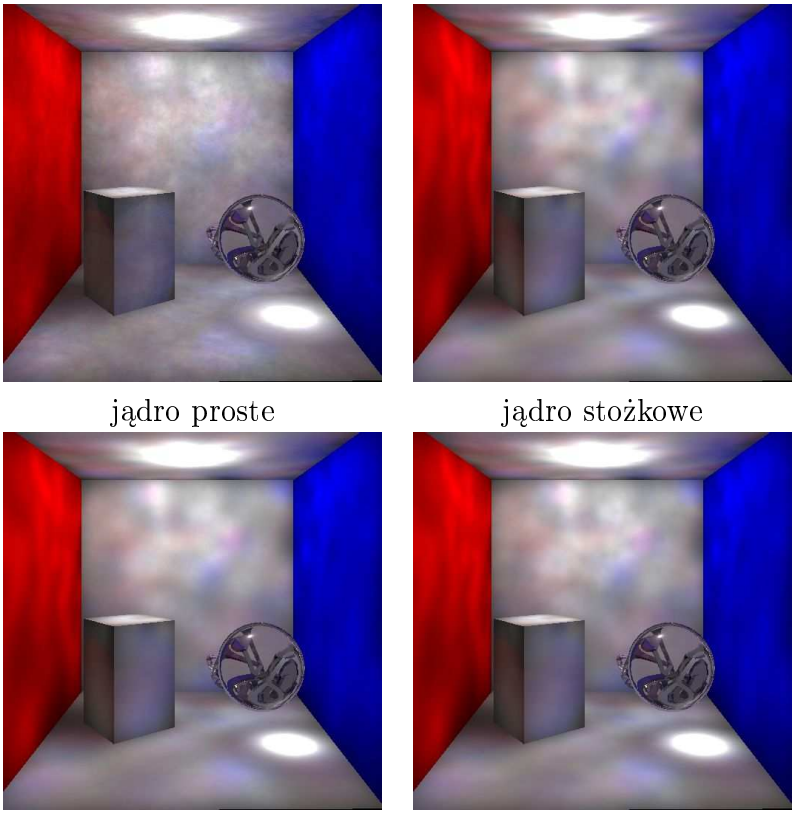
Tablice 5.7 i 5.8 pokazują porównanie szybkości renderingu obrazów z użyciem różnych jąder.

Kernele stożkowy, gaussowski oraz Epanechnikova dają bardzo zbliżone rezultaty. Pod względem szybkości i jakości najlepszy jest kernel Epanechnikova. Kernel Epanechnikova, Gaussa oraz stożkowy powodują, że kaustyki i cienie są wyraźniejsze niż przy użyciu prostego kernela, który powoduje większe rozmycie szczegółów.

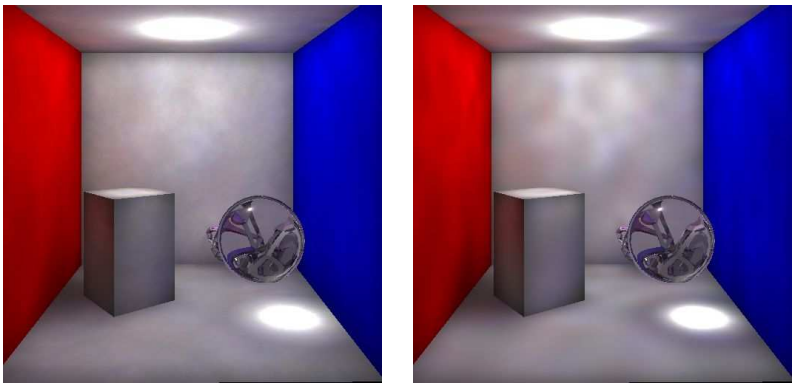
3.7 Estymacja z filtrowaniem czasowym w animacjach

3.7.1 Filtry proste

Mapy fotonów umożliwiają dokonanie estymacji radiancji $L(x, \omega_i)$ dla danego punktu x, kierunku i. W celu zwiększenia dokładności estymacji lub w celu zwiększenia szybkości renderingu bierze się pod uwagę wartości $L(x, \omega_i)$ dla sąsiednich klatek, jeżeli

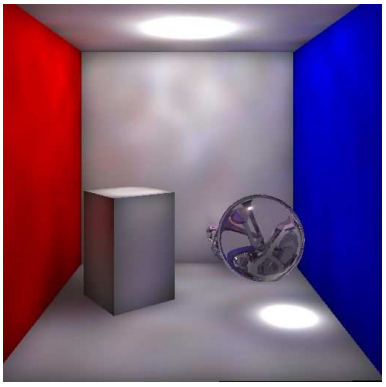


Rysunek 3.5: Porównanie efektów działania poszczególnych jąder. Użyto 10000 fotonów dla mapy globalnej, 3000 dla mapy kaustyk, 50 fotonów w estymacji dla mapy globalnej i 50 dla mapy kaustyk.

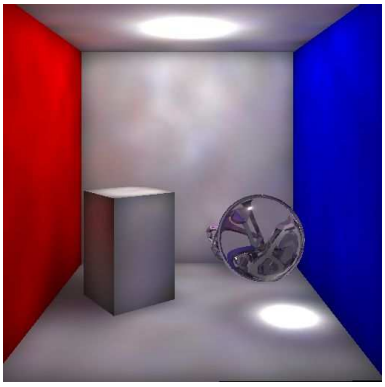


jądro proste

jądro stożkowe

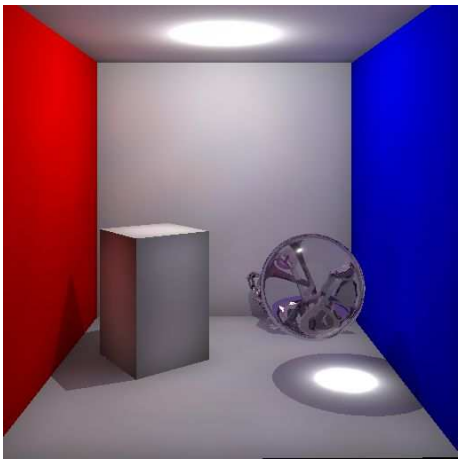


jądro gaussowskie

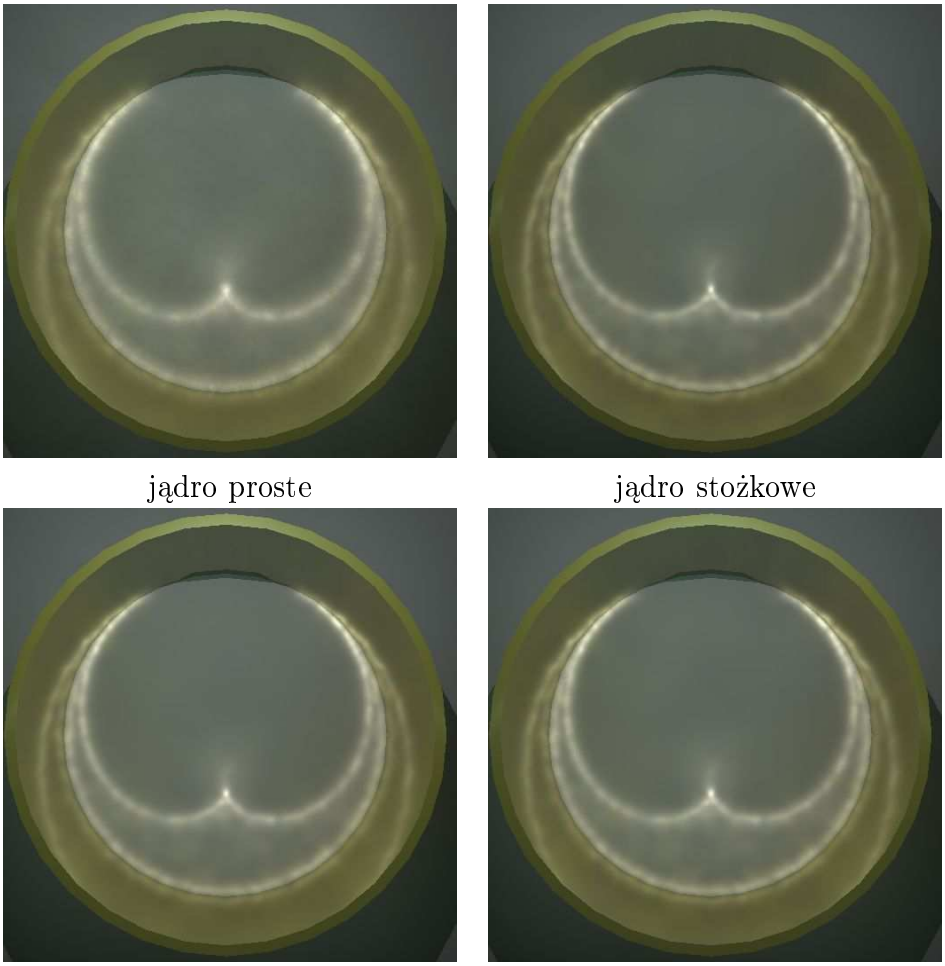


jądro Epanechnikova

Rysunek 3.6: Porównanie efektów działania poszczególnych jąder. Użyto 50000 fotonów dla mapy globalnej, 3000 dla mapy kaustyk, 250 fotonów w estymacji dla mapy globalnej i 50 dla mapy kaustyk.



Rysunek 3.7: Połączenie oświetlenia bezpośredniego przez punktowe światło z estymacją oświetlenia pośredniego z mapy fotonów. Użyto 50000 fotonów dla mapy globalnej, 3000 dla mapy kaustyk, 500 fotonów w estymacji dla mapy globalnej i 50 dla mapy kaustyk.



Rysunek 3.8: Porównanie efektów działania poszczególnych jąder w filtrowaniu kaustyk o ostrych krawędziach. Użyto 5000 fotonów dla mapy globalnej, 50000 dla mapy kaustyk, 100 fotonów w estymacji dla mapy globalnej i 100 dla mapy kaustyk.

oświetlenie nie uległo znaczącej zmianie. Estymacja $L(x, \omega_i)$ z użyciem filtrowania czasowego wyrażona jest wzorem:

$$L(x, \omega_i, 0) = \sum_{j=-M}^{K} w_j L(x, \omega_i, j) \,, \tag{3.18}$$

gdzie $L(x, \omega_i, 0)$ to wartość $L(x, \omega_i)$ w bierzącej klatce, natomiast $L(x, \omega_i, j)$ dla $j \neq 0$ to wartość $L(x, \omega_i)$ w poprzednich (j<0) i następnych (j>0) klatkach. M to ilość filtrowanych klatek w tył, a K w przód. Wartości w_j są wagami i sumują się do 1. Rozmiarem filtra nazywamy liczbę $K + M + 1$. Wagi w_j czyli kształt filtru obliczany jest z dowolnej funkcji ciągłej *f*. Dziedzina funkcji f jest ograniczona do obszaru [-1,1]. System używa następujących filtrów:

- Filtr prosty $f(x) = 0.5$
- Filtr trójkątny $f(x) = -x + 1$ dla $x \geq 0$ oraz $f(x) = x + 1$ dla $x < 0$
- Filtr gaussowski $f(x) = \frac{4}{\sqrt{2*\pi}} \exp(-8x^2)$
- Filtr Epanechnikova $f(x) = 0.75 * (x + 1) * (x - 1)$

Niech $N = 2\max(K, M) + 1$. Wyznaczany jest zbiór wag używając następującego wzoru w celu uzyskania współczynników:

$$w_j^* = f(\frac{2j}{N})\frac{2}{N}$$
$$w_j = \frac{w_j^*}{\sum_{k=-M}^{K} w_k} \tag{3.19}$$

Powyższe wzory stosuje się dla $j \in [-M, K]$. W ten sposób, gdy z jednej strony mamy krótszy przedział filtrowania spowodowany końcem lub początkiem animacji, współczynniki po lewej i po prawej stronie bierzącej klatki są takie same na przedziale [-min(M,K),min(M,K)].

3.7.2 Zmienny rozmiar filtra

Przy stałym rozmiarze filtra gwałtowne zmiany oświetlenia przenoszą się na sąsiednie klatki powodując zwiększenie błędu estymacji. Dopasowywanie rozmiaru filtra polega na początkowym ustawieniu rozmiaru filtra na 1 (tylko bierząca klatka animacji) i na stopniowym zwiększaniu rozmiaru w przód i w tył dla klatek j spełniających następujące kryterium:

$$|L(x, \omega_i, j) - L(x, \omega_i, 0)| < \alpha |L(x, \omega_i, 0)| \tag{3.20}$$

Rozmiar filtra jest dopasowywany dla każdego rozpatrywanego punktu x w scenie. Gdy kryterium 3.20 nie jest spełnione dla danej klatki j, filtr nie ulega dalszemu rozszerzaniu w tą stronę, oraz rozpatrywana klatka nie bierze udziału w filtrowaniu. Parametr $\alpha > 0$ można dobierać dowolnie, jednak w praktyce wartość 0.333 dobrze się sprawdziła. Bardzo duże wartości tego parametru pozwalają zilustrować artefakty powstające przy dużej zmienności oświetlenia i za dużym rozmiarze filtra.

3.7.3 Bilateralne filtry czasowe

Filtr bilateralny umożliwia nadanie mniejszych wag dla radiancji w punkcie x w klatkach, gdzie są duże wahania radiancji, dzięki temu wartość obliczanej radiancji jest bardziej odporna na gwałtowne zmiany oświetlenia. Filtrowanie bilateralne polega na wyznaczeniu wartości:

$$b_j^* = w_j[1 - \frac{d(j)}{d_{max}}]$$

$$b_j = \frac{b_j^*}{\sum_{k=-M}^{K} b_k^*}\,, \tag{3.21}$$

gdzie $d(j) = |L(x, \omega_i, j) - L(x, \omega_i, 0)|$, a d_{max} to wartość maksymalna tej różnicy na filtrowanym przedziale. Wartoć $d(j)$ jest odległością euklidesową między $L(x, \omega_i, j)$ i $L(x, \omega_i, 0)$. Wartości są wektorami 3 kanałów R, G, B.

Teoretycznie można brać pod uwagę wszystkie klatki animacji w czasie filtrowania, ale duże rozmiary map fotonów oraz duże ilości klatek powodują, że nie wystarcza na to pamięci operacyjnej, oraz czas działania byłby o wiele dłuższy.

3.7.4 Polepszenie jakości filtrowania czasowego

Eksperymenty wykazały, że przy dużym rozmiarze filtra poruszające się kaustyki są zbyt mocno rozmywane. Aby wyeliminować ten efekt zamiast wzoru

$$b_j^* = w_j[1 - \frac{d(j)}{d_{max}}] \tag{3.22}$$

zastosowano wzór

$$b_j^* = w_j \left(1 - \left(\frac{d(j)}{d_{max}}\right)^2\right) \tag{3.23}$$

powodujący przypisywanie mniejszych wag dla większych różnic odległości. Problem i poprawienie ilustruje rysunek 3.9.

3.7.5 Filtrowanie czasowe i metoda final gathering

Możliwe jest połączenie metody final gathering z filtrowaniem czasowym na następujące sposoby:

1. Filtrowanie czasowe radiancji obliczonej za pomocą final gathering.
2. Filtrowanie czasowe radiancji estymowanej bezporednio z mapy fotonów, użyte dla każdej próbki w metodzie final gathering.
3. Połączenie powyższych metod.

Połączenie filtrowania temporalnego i final gathering stosuje się w przypadku małej ilości fotonów na jedną klatkę animacji i przy estymacji z użyciem małej ilości fotonów. Metoda final gathering wymaga brania pod uwagę 200-1000 próbek dla

(a) (b) (c)

Rysunek 3.9: Poprawienie jakości filtrowania czasowego poprzez zastosowanie filtru 3.23. Rysunek (a) nie był filtrowany czasowo. Rysunek (b) jest wyrenderowany z użyciem filtra czasowego dla $M = 0$, $K = 8$ (wzór 3.18) i o wagach danych wzorem 3.22. Rysunek (c) jest wyrenderowany z użyciem filtra czasowego dla $M = 0$, $K = 8$ i o wagach danych wzorem 3.23. Na rysunku (b) kaustyki zostały za bardzo rozmyte, na rysunku (c) są o wiele wyraźniejsze.

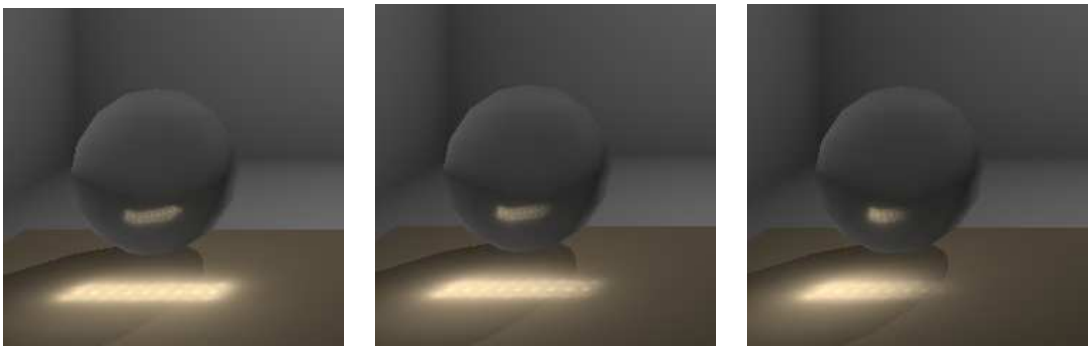
powierzchni rozpraszających. Każda próbka wymaga sprawdzenia przecięcia promienia ze sceną, co w przypadku ray tracingu dla scen dynamicznych jest często bardziej czasochłonne niż zwiększenie liczby fotonów na jedną klatkę i zwiększenie liczby fotonów w estymacji. Większych rozmiarów filtrów czasowych i dużych map fotonów może nie starczyć pamięci operacyjnej dla wielu map fotonów. W takim przypadku można stosować metodę final gathering z filtrowaniem czasowym dla uzyskania lepszych efektów. Opisywany system pozwala zastosować metodę (2), gdyż użycie metody (1) można zasymulować poprzez zwiększenie liczby próbek. Metody (1) i (3) czasem będą przypisywać małe wagi dla radiancji obliczonej kosztowną metodą final gathering, co nie jest efektywnym rozwiązaniem.

3.7.6 Porównanie filtrów czasowych

Rysunki 3.10 i 3.11 przedstawia scenę caustic_tf.3ds z przesuwającą się szklaną kulą a wraz z nią kaustyką. Przy renderingu tej sceny używano 16000 fotonów w mapie globalnej, 220 fotonów do estymacji, 3000 fotonów w mapie kaustyk, 160 do estymacji.

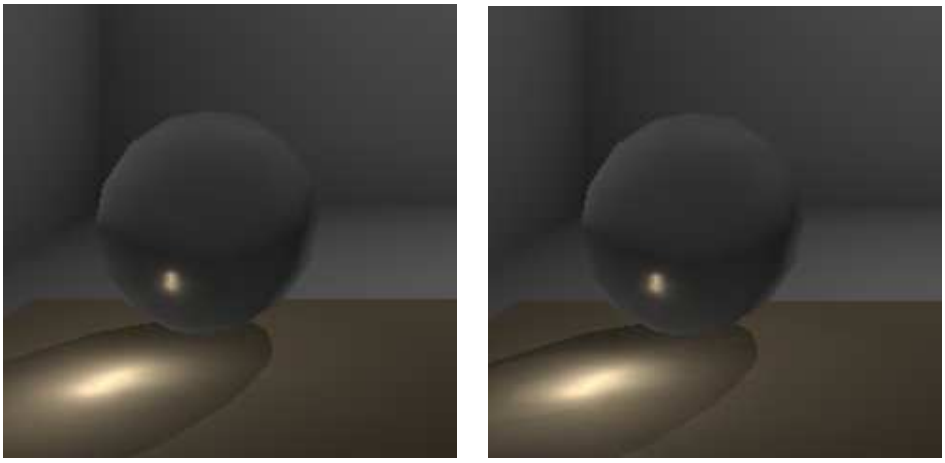
3.7.7 Wnioski

Przykładowe animacje znajdują się w katalogu *avi/temporal_filtering/* na płycie CD dołączonej do tego dokumentu. W przypadku animacji stosowanie bilateralnych filtrów temporalnych znacznie zmniejsza fluktuacje oświetlenia w przypadku bezpośredniej estymacji radiancji z mapy fotonów. Przykładowe filmy cor2_t0.avi, movement_t0.avi (bez filtrowania temporalnego) i cor2_t1.avi, movment_t1_b1_21.avi (z bilateralnym filtrowaniem temporalnym) pokazują różnice. Fluktuacje w filmie cor2_t1.avi, movment_t1_b1_21.avi są praktycznie niezauważalne.



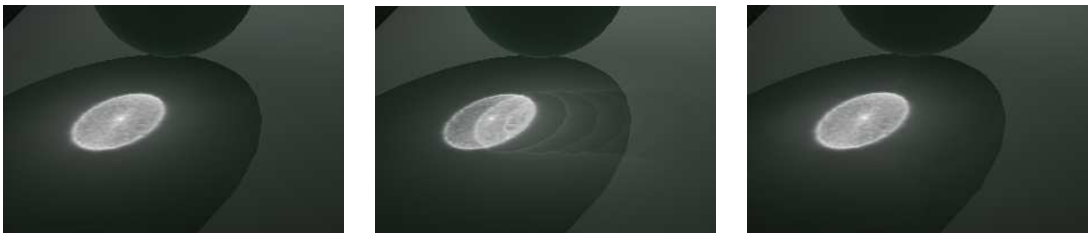
(a) (b) (c)

Rysunek 3.10: Porównanie filtrów czasowych. Filtr zwykły (a), stożkowy (b) i gaussowski (c).



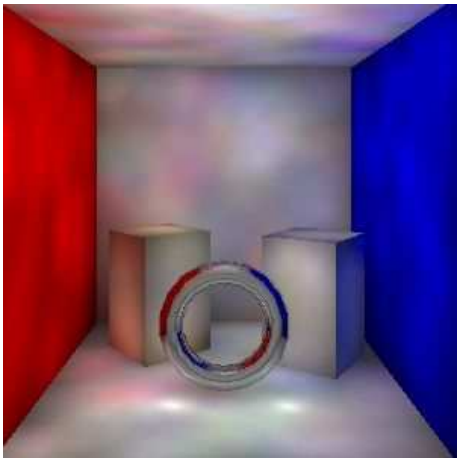
(a) (b)

Rysunek 3.11: Porównanie filtrów czasowych. Obraz wyrenderowany z użyciem fotonów z jednej klatki (a) oraz z użyciem bilateralnego filtrowania czasowego (b).

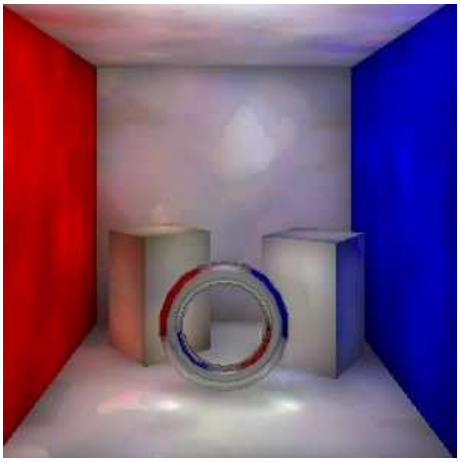


(a) (b) (c)

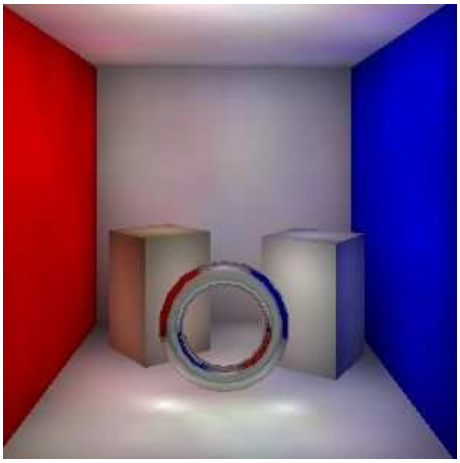
Rysunek 3.12: Filtrowanie czasowe kaustyki o ostrych krawędziach. Obraz (a) jest wyrenderowany z użyciem fotonów z jednej klatki. Obraz (b) i (c) jest wyrenderowany z użyciem stożkowych bilateralnych filtrów czasowych, różnica jest tylko w parametrze α z równania 3.20. Obraz (b) wyrenderowano z $\alpha = 2.333$, natomiast obraz (c) z $\alpha = 0.333$.



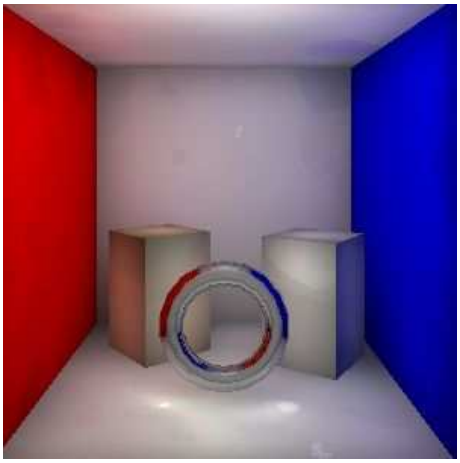
n=6000 k=50



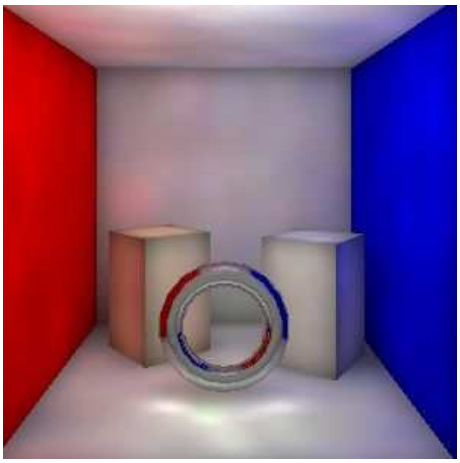
n=6000 k=50



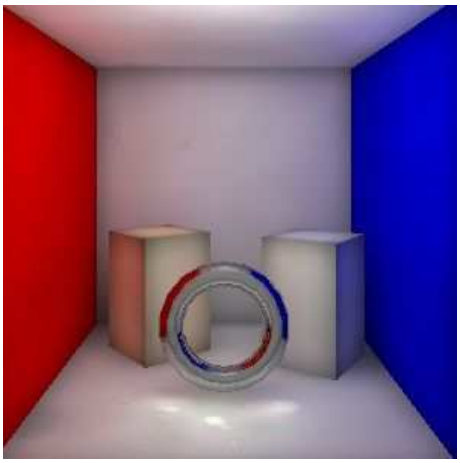
n=10000 k=200



n=10000 k=200



n=50000 k=350



n=50000 k=350

Rysunek 3.13: Porównanie filtrów czasowych. Lewa kolumna przedstawia obrazy wyrenderowane z użyciem fotonów z jednej klatki. Prawa kolumna zawiera obrazy wyrenderowane z użyciem bilateralnych filtrów czasowych o kształcie stożkowym dla $M = 0$ i $K = 5$ (wzór 3.18). Scena cor2.3ds, n fotonów w mapie, k użytych do estymacji dla map globalnych, 2000 fotonów, 40 użytych do estymacji.

Rysunek 3.14: Porównanie filtrów temporalnych. Lewa kolumna: $M \in \{0...3\}$ i $K = 3$. Prawa kolumna: $M \in \{0..10\}$ i $K = 10$.

Plik movement_t1_b0_7.avi został wyrenderowany używając maksymalnego rozmiaru filtra 7 oraz bez użycia filtru bilateralnego (2000 fotonów na klatkę). Plik movement_t1_b1_21.avi został wyrenderowany używając maksymalnego rozmiaru filtra 21, przy 3000 fotonach na klatkę.

Rozdział 4

Sposób użycia opisywanego systemu do ray tracingu

4.1 Uruchomienie

4.1.1 Rozpoczęcie renderingu

System uruchamiany jest z linii poleceń w następujący sposób:

```
./r <file.3ds>
```

Wywołanie to spowoduje rozpoczęcie procesu renderingu animacji zawartej w pliku file.3ds. Każda klatka animacji po wyrenderowaniu zostaje zapisana do katalogu `file` w pliku .hdr o wysokim zakresie dynamicznym w formacie RGBE.

4.1.2 Mapowanie tonów

Po skończeniu renderingu pliki .hdr są konwertowane do plików .jpg . Podczas konwersji uruchamiany jest zewnętrzny program do mapowania tonów. Mapowanie tonów polega na transformacji obrazu o wysokim zakresie dynamicznym do obrazu w niskim zakresie dynamicznym (256 wartości dla każdego kanału R,G,B). Pod Windows wykorzystany jest plugin do programu HDRShop (tonemap.exe), natomiast pod Linuxem wykorzystane są programy z pakietów pfstools-1.1 oraz pfstmo-1.0. Programy te mają nazwy pfsinrgbe, pfsoutppm, pfstmo_drago03. Przekształcanie plików .hdr na pliki .jpg realizuje się za pomocą wywołania

```
./r -t <file>
```

gdzie <file> jest nazwą katalogu z poszczególnymi klatkami animacji.

4.1.3 Szybki podgląd

System pozwala na dokonanie podglądu animacji za pomocą biblioteki OpenGL. Podgląd w OpenGL jest szybszy niż metoda śledzenia promieni dla małych ilości trójkątów. Wywołanie

```
./r -p <file.3ds>
```

spowoduje otworzenie okna i wyświetlenie w nim animacji.

4.2 Pliki konfiguracyjne

System używa dwóch typów plików konfiguracyjnych:

- Plik rozszerzający format 3DS
- Plik definiujący opcje syntezy animacji

4.2.1 Plik rozszerzający format 3DS

Plik rozszerzający format 3DS pozwala na zdefiniowanie świateł płaszczyznowych oraz przedefianiowanie paramterów materiału. Plik ma nazwę podstawową taką jak plik .3ds, i rozszerzenie .3de. W tym pliku można zdefiniować światło płaszczyznowe wprowadzając linię:

```
lom <Material Name> <R> <G> <B> <Intensity>
```

Kolor światła wyrażony jest w watach na jednosktę powierzchni i wynosi $X*Intensity$ dla $X \in R, G, B$. Możliwe jest przedefiniowanie współczynników materiału za pomocą następującej komendy:

```
mat <MaterialName>
  kd <R> <G> <B>
  ks <R> <G> <B>
  kt <R> <G> <B>
  kr <R> <G> <B>
  ri <Refraction Index>
```

Pozwala to na przypisaniu materiałowi o nazwie MaterialName współczynników odbicia rozproszonego (k_d), współczynnika odbicia zwierciadlanego (k_s) w zmodyfikowanym modelu Phonga, współczynnika przeźroczystości (k_t), współczynnika odbicia idealnie zwierciadlanego (k_r) oraz współczynnika załamania światła (r_i).

4.2.2 Plik konfiguracyjny systemu

Plik definiujący opcje syntezy animacji zawiera szereg komend i wartości postaci:

```
<komenda> <wartość>
```

gdzie pole `<wartość>` jest liczbą lub łańcuchem znaków, a pole `<komenda>` jest łańcuchem znaków. Plik nazywa się default.cfg i zostaje wczytany z bierzącego katalogu. Po wczytaniu pliku default.cfg następuje próba wczytania pliku o takiej samej nazwie bazowej jak nazwa pliku .3ds, ale z rozszerzeniem .cfg z katalogu zawierającego plik .3ds. Dzięki temu można przedefiniować parametry specyficzne dla danej sceny. Komendy mogą występować w dowolnej kolejności. Gdy system nie znajdzie pliku .cfg lub gdy nie są zdefiniowane wszystkie opcje, przyjmowane są wartości domyślne dla niezdefiniowanych opcji. Plik .cfg może zawierać podzbiór niżej wymienionych komend (po komendzie wymieniona jest wartość domyślna):

- **Metoda do syntezy animacji**

  ```
  render_mode RM_MONTE_CARLO_FAST
  ```

Poniżej są opisane metody zaimplementowane w systemie:

typ	opis
RM_DIRECT_LIGHT	tylko oświetlenie bezpośrednie, brak kaustyk
RM_MONTE_CARLO_FAST	oświetlenie bezpośrednie, oświetlenie pośrednie z mapy fotonów, kaustyki
RM_MONTE_CARLO	oświetlenie bezpośrednie, Final Gathering, kaustyki, odbicie zwierciadlane zgodne z BRDF

- **Rozdzielczość animacji**

  ```
  res_x 512
  res_y 512
  ```

- **Głębokość rekursji dla odbić idealnie zwierciadlanych**

  ```
  recursion 3
  ```

- **Ilość próbek na jedno światło płaszczyznowe**

  ```
  shadow_ray_samples 10
  ```

- **Współczynnik załamania światła**

  Wartość domyślna, może być przedefiniowana w pliku rozszerzającym format .3ds.

  ```
  refraction_index 1.5
  ```

- **Korekcja gamma**

  Wartość używana w czasie mapowania tonów.

  ```
  gamma_correction 2.2
  ```

- **Typ odbicia fotonu podczas generowania kaustyk**

  ```
  caustic_reflection CR_MIRROR
  ```

  Poniżej przedstawione są możliwe typy odbicia fotonu:

typ	opis
CR_MIRROR	odbicie idealnie zwierciadlane
CR_BRDF	odbicie zgodne z funkcją BRDF

- **Maksymalny początkowy promień koła dla algorytmu znajdowania najbliższych fotonów**

  Odpowiednie dobranie tej wartości może znacząco wplynąć na szybkość syntezy obrazu, ponieważ podczas poszukiwania najbliższych fotonów mogą być odrzucane gałęzie kd-drzewa, które znajdują się poza kulą o podanym promienu.

  ```
  max_search_dist 100.0
  ```

- **Ilość fotonów w globalnej mapie fotonów**

  ```
  max_photon_count 20000
  ```

- **Ilość fotonów do estymacji dla globalnej mapy fotonów**

  ```
  estimate_photon_count 300
  ```

- **Ilość fotonów w kaustycznej mapie fotonów**

  ```
  max_caustic_photon_count 20000
  ```

- **Ilość fotonów do estymacji dla kaustycznej mapy fotonów**

  ```
  estimate_caustic_photon_count 50
  ```

- **Ilość próbek dla metody Final Gathering**

  ```
  final_gather_diffuse_samples 6
  ```

  Ilość próbek każdego z dwóch kierunków parametryzacji sferycznej dla metody Final Gathering. Faktyczna ilość próbek będzie zatem kwadratem podanej liczby.

- **Ilość próbek dla obliczania odbicia zwierciadlanego zgodnie z BRDF**

  ```
  final_gather_specular_samples 32
  ```

- **Typ jądra przestrzennego**

  ```
  kernel_type KERNEL_EPANECHNIKOV
  ```

Poniżej przedstawione są możliwe typy jądra:

typ	opis
KERNEL_SIMPLE	jądro proste
KERNEL_CONE	jądro stożkowe
KERNEL_GAUSS	jądro gausowskie
KERNEL_EPANECHNIKOV	jądro Epanechnikova

- **Typ filtra czasowego**

  ```
  temporal_filter_type TEMPORAL_FILTER_NONE
  ```

  Poniżej przedstawione są możliwe typy filtra czasowego:

typ	opis
TEMPORAL_FILTER_NONE	brak filtrowania czasowego
TEMPORAL_FILTER_SIMPLE	prosty filtr czasowy
TEMPORAL_FILTER_CONE	stożkowy filtr czasowy
TEMPORAL_FILTER_GAUSS	gaussowski filtr czasowy

- **Rozmiar filtra czasowego**

  Maksymalna ilość klatek używanych do filtrowania czasowego przy estymacji z mapy fotonów.

  ```
  temporal_filter_size 7
  ```

- **Współczynnik alfa dla filtra czasowego**

  Współczynnik określający warunek zatrzymania procedury rozszerzania filtra czasowego, w przypadku wykrycia dużych zmian oświetlenia w sąsiednich klatkach. Dokładniejszy opis znajduje się w rozdziale 3, w podrozdziale o filtrowaniu czasowym.

  ```
  temporal_filter_alpha 0.33
  ```

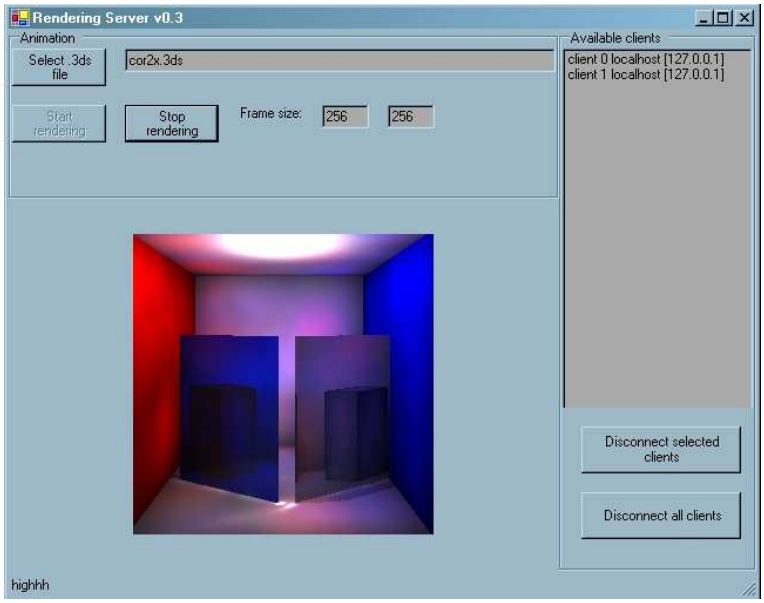
- **Filtr bilateralny**

  Wartość 1 dla parametru powoduje użycia bilateralnego filtra czasowego, wartość 0 oznacza nie używanie filtra bilateralnego.

  ```
  bilateral 0
  ```

4.3 Rendering rozproszony

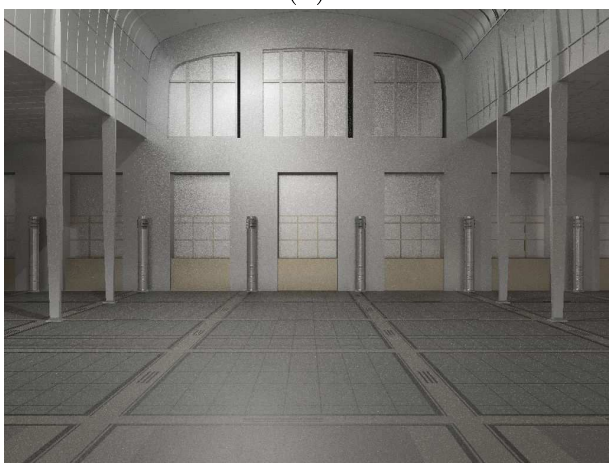
System może być wykorzystany w renderingu w sieci TCP/IP. Katalog *src/rendering_farm* na dołączonej płycie CD zawiera przykładowy program napisany w C# do renderingu rozposzonego. Program składa się z serwera, z którego wysyłamy polecenia do klientów. Klient to program, który wykorzystuje system do ray tracingu do renderingu poszczególnych linni obrazu. Po połączeniu z serwerem klient otrzymuje kolejne polecenia i odsyła wyrenderowane linie obrazu serwerowi. Rysunek 4.1 przedstawia zrzut ekranu programu serwera.



Rysunek 4.1: Program Rendering Server.

(a)

(b)



(c)

Rysunek 4.2: Scena wyrenderowana przy użyciu metod RM_DIRECT_LIGHT (a), RM_MONTE_CARLO_FAST (b), RM_MONTE_CARLO (c).

(a)

(b)

(c)

Rysunek 4.3: Scena wyrenderowana przy użyciu metod RM_DIRECT_LIGHT (a), RM_MONTE_CARLO_FAST (b), RM_MONTE_CARLO (c).

Rozdział 5

Porównanie szybkości metod zaimplementowanych w systemie

Wszystkie testy są wykonane na komputerze z procesorem Athlon XP 2.6 (2000 Mhz), 256 Mb RAM pod systemem Linux i z użyciem kompilatora GCC 3.3.1. Czas renderingu nie uwzględnia czasu budowania map fotonów.

5.1 Metoda śledzenia promieni

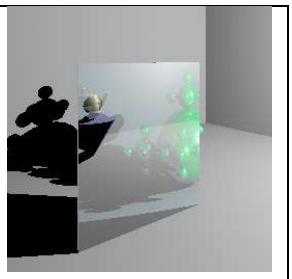
Wszystkie obrazy zostały wyrenderowane w rozdzielczości 256x256, przy głębokości rekursji 3 dla promieni odbitych w sposób idealnie zwierciadlany. Tablice 5.1 i 5.2 przedstawiają charakterystyki renderowanych scen oraz szybkości renderingu.

5.2 Metoda map fotonów z bezpośrednią estymacją radiancji

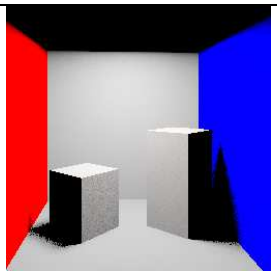
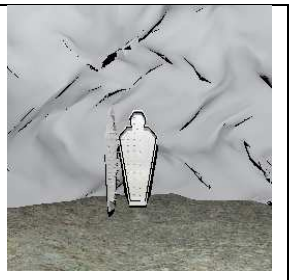
Wszystkie obrazy zostały wyrenderowane w rozdzielczości 256x256, przy głębokości rekursji 3 dla promieni odbitych w sposób idealnie zwierciadlany. Niech n oznacza ilość fotonów w mapie, a k - ilość fotonów w estymacji. Dla globalnej mapy fotonów $n = 30000$, $k = 300$. Dla kaustycznej mapy fotonów $n = 20000$, $k = 50$. Tablice 5.3 i 5.4 przedstawiają charakterystyki renderowanych scen oraz szybkości renderingu. Czas budowy map fotonów dla sceny ss.3ds jest bardzo długi w porównaniu z czasami dla innych scen. Scena ss.3ds jest otwarta, więc wiele fotonów po pierwszym odbiciu ucieka w pustą przestrzeń. Metoda bezpośredniej estymacji radiancji używana jest do obliczania oświetlenia pośredniego, więc fotony po pierwszym odbiciu nie są zapamiętywane.

5.3 Metoda map fotonów z final gathering

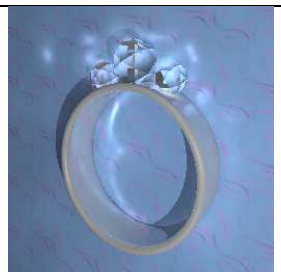
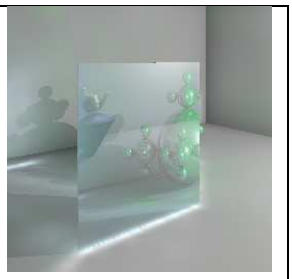
Wszystkie obrazy zostały wyrenderowane w rozdzielczości 256x256, przy głębokości rekursji 3 dla promieni odbitych w sposób idealnie zwierciadlany. Niech n oznacza

			
nazwa sceny	sp2.3ds	ring4.3ds	ref3a.3ds
ilość wierzchołków	554	1638	15634
ilość trójkątów	1048	3254	31096
ilość obiektów	4	6	5
ilość świateł	1	1	1
czas renderingu (sek.)	1.55	2.88	7.12
ilość przecięć promień-trójkąt na sek.	5373904	5610516	6400399

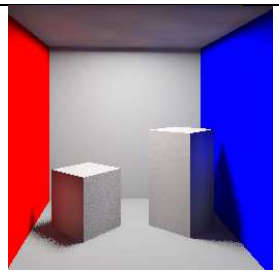
Tablica 5.1: Test szybkości dla klasycznej metody śledzenia promieni.

			
nazwa sceny	cornell6.3ds	ss.3ds	cave.3ds
ilość wierzchołków	44	2562	18154
ilość trójkątów	38	5082	33188
ilość obiektów	9	4	9
ilość świateł	1 $\diamond$	2 $\diamond$	1
ilość próbek na światło	20	20	1
czas renderingu (sek.)	10.36	46.67	63.37
ilość przecięć promień-trójkąt na sek.	4780342	8118446	4394225

Tablica 5.2: Test szybkości dla klasycznej metody śledzenia promieni. Symbol $\diamond$ w rubryce "ilość świateł" oznacza, że w scenie znajdują się płaszczyznowe źródła światła.

			
nazwa sceny	sp2.3ds	ring4.3ds	ref3a.3ds
ilość wierzchołków	554	1638	15634
ilość trójkątów	1048	3254	31096
ilość obiektów	4	6	5
ilość świateł	1	1	1
czas renderingu (sek.)	5.76	6.03	15.29
czas budowy map fotonów (sek.)	1.33	2.29	18.84
ilość przecięć promień-trójkąt na sek.	1446103	2679649	2980434

Tablica 5.3: Test szybkości dla metody bezpośredniej estymacji radiancji z mapy fotonów z uwzględnieniem oświetlenia bezpośredniego obliczonego klasyczną metodą śledzenia promieni.

			
nazwa sceny	cornell6.3ds	ss.3ds	cave.3ds
ilość wierzchołków	44	2562	18154
ilość trójkątów	38	5082	33188
ilość obiektów	9	4	9
ilość świateł	1 $\diamond$	2 $\diamond$	1
ilość próbek na światło	20	20	1
czas renderingu (sek.)	15.47	54.20	73.43
czas budowy map fotonów (sek.)	1.33	62.55	135.8
ilość przecięć promień-trójkąt na sek.	3201314	6991235	3792211

Tablica 5.4: Test szybkości dla metody bezpośredniej estymacji radiancji z mapy fotonów z uwzględnieniem oświetlenia bezpośredniego obliczonego klasyczną metodą śledzenia promieni. Symbol $\diamond$ w rubryce "ilość świateł" oznacza, że w scenie znajdują się płaszczyznowe źródła światła.

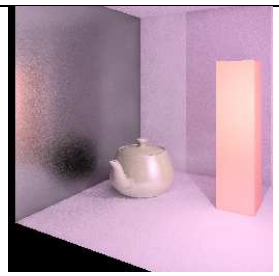
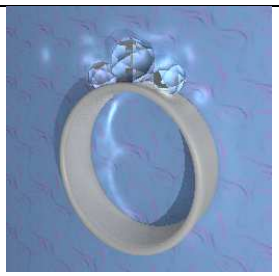
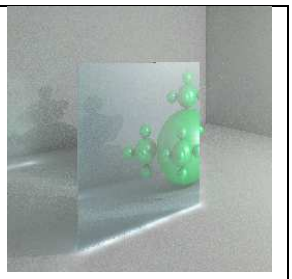
ilość fotonów w mapie, a k - ilość fotonów w estymacji. Dla globalnej mapy fotonów $n = 30000$, $k = 300$. Dla kaustycznej mapy fotonów $n = 20000$, $k = 50$. Ilość próbek dla final gathering wynosi 49 dla powierzchni rozpraszających oraz 32 dla powierzchni z połyskiem. Tablice 5.3 i 5.4 przedstawiają charakterystyki renderowanych scen oraz szybkości renderingu.

5.4 Metoda map fotonów z filtrowaniem przestrzennym

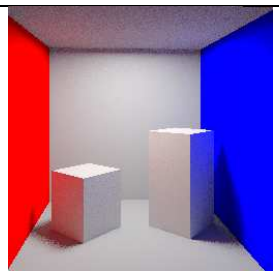
Tablice 5.7 i 5.8 przestawiają szybkości renderingu scen cor2.3ds i ring3.3ds z wykorzystaniem różnych jąder w filtrowaniu przestrzennym radiancji z map fotonów.

5.5 Metoda map fotonów z filtrowaniem czasowym

Tablica 5.10 przedstawia szybkości renderingu z wykorzystaniem czasowych bilateralnych filtrow stożkowych o zmiennym rozmiarze. Rozmiar obrazu wynosi 128x128, globalna mapa fotonów: $n = 8000$, $k = 150$, kaustyczna mapa fotonów: $n = 2000$, $k = 32$. Parametry scen są przedstawione w tablicy 5.9.

			
nazwa sceny	sp2.3ds	ring4.3ds	ref3a.3ds
ilość wierzchołków	554	1638	15634
ilość trójkątów	1048	3254	31096
ilość obiektów	4	6	5
ilość świateł	1	1	1
czas renderingu (sek.)	174.97	228.68	422.14
czas budowy map fotonów (sek.)	0.94	1.84	18.23
ilość przecięć promień-trójkąt na sek.	972945	1750629	2130687

Tablica 5.5: Test szybkości dla metody Final Gathering z uwzględnieniem oświetlenia bezpośredniego obliczonego klasyczną metodą śledzenia promieni.

			
nazwa sceny	cornell6.3ds	ss.3ds	cave.3ds
ilość wierzchołków	44	2562	18154
ilość trójkątów	38	5082	33188
ilość obiektów	9	4	9
ilość świateł	1 $\diamond$	2 $\diamond$	1
ilość próbek na światło	20	20	1
czas renderingu (sek.)	120.42	101.21	1038.22
czas budowy map fotonów (sek.)	0.28	0.34	36.27
ilość przecięć promień-trójkąt na sek.	1370858	7376907	3721933

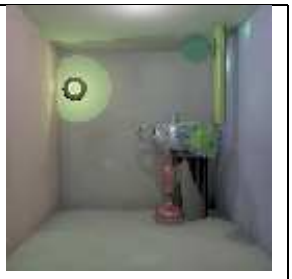
Tablica 5.6: Test szybkości dla metody Final Gathering z uwzględnieniem oświetlenia bezpośredniego obliczonego klasyczną metodą śledzenia promieni. Symbol $\diamond$ w rubryce "ilość świateł" oznacza, że w scenie znajdują się płaszczyznowe źródła światła.

Jądro	Czas renderingu obrazu (sek.)	Ilość przecięć promień-trójkąt na sekunde
proste	5.16	2491369
stożkowe	5.50	2337357
gaussowskie	6.23	2063478
Epanechnikova	5.47	2350177

Tablica 5.7: Porównanie szybkości. Scena cor3.3ds, rozmiar obrazu 256x256, 25000 fotonów w mapie globalnej, 5000 fotonów w mapie kaustyk, estymacja na 200 fotonach dla mapy globalnej, 50 dla mapy kaustyk.

Jądro	Czas renderingu obrazu (sek.)	Ilość przecięć promień-trójkąt na sekunde
proste	9.42	2520017
stożkowe	9.78	2427256
gaussowskie	10.71	2216486
Epanechnikova	9.81	2419823

Tablica 5.8: Porównanie szybkości. Scena ring3.3ds, rozmiar obrazu 256x256, 5000 fotonów w mapie globalnej, 50000 fotonów w mapie kaustyk, estymacja na 100 fotonach dla mapy globalnej, 100 dla mapy kaustyk.

			
nazwa pliku	cor2.3ds	caustic_tf.3ds	move.3ds
ilość trójkątów	360	248	2872
ilość obiektów	6	3	11
ilość świateł	1	1	2

Tablica 5.9: Parametry scen.

scena	K	Czas renderingu (sek.)	Ilość przecięć promień-trójkąt na sekunde.
cor2.3ds	0	0.81	1980575
	1	1.40	1145904
	2	1.99	806164
	4	3.09	519180
	8	4.97	322790
	16	9.45	169764
move.3ds	0	2.23	5276844
	1	2.89	4071751
	2	3.39	3471199
	4	4.28	2749383
	8	6.25	1882777
	16	9.69	1214382
caustic_tf.3ds	0	0.94	1382532
	1	1.66	782880
	2	2.30	565035
	4	3.43	378886
	8	5.88	221017
	16	9.79	132745

Tablica 5.10: Test szybkości renderingu z filtrowaniem czasowym.

Bibliografia

[Shirley2000] Peter Shirley – "Realistic Ray Tracing", A.K. Peters 2000

[Dutre2003] Philip Dutre – "Global Illumination Compendium", http://www.cs.kuleuven.ac.be/~phil/GI/

[Jensen2001] Per H. Christensen, Henrik Wann Jensen, Frank Suykens – "A Practical Guide to Global Illumination using Photon Mapping" (Siggraph 2001 course 38)

[Jensen1996] Henrik Wann Jensen – "Global Illumination using Photon Maps", Eurographics 1996

[Havran2000] V. Havran – "Heuristic Ray Shooting Algorithms", praca doktorska, Praga 2000

[Cohen1993] M. F. Cohen, J. R. Wallace – "Radiosity and Realistic Image Synthesis", A. P. 1993

[Łukaszewski2001] Andrzej Łukaszewski – "Offsets and Minkowski operators for speeding up global illumination methods", praca doktorska , 2001.

[TomMand1998] C. Tomasi, R. Manduchi – "Bilateral Filtering for Gray and Color Images", IEEE Computer Society Washington, DC, USA 1998

[Foley1995] J. D. Foley, A. van Dam, S. K. Feiner – "Wprowadzenie do grafiki komputerowej", WNT, Warszawa 1995

[RedBook] Jackie Neider, Tom Davis, and Mason Woo – "Open GL: The Red Book", Addison-Wesley Publishing Company, 1994

[BlueBook] "Open GL: The Blue Book", Addison-Wesley Publishing