Przemysław MAZUREK
ZACHODNIOPOMORSKI UNIWERSYTET TECHNOLOGICZNY W SZCZECINIE, KATEDRA PRZETWARZANIA SYGNAŁÓW I INŻYNIERII MULTIMEDIALNEJ,
26. Kwietnia 10, 71-126 Szczecin

Estymacja lakunarności obiektu 2D z wykorzystaniem GPGPU

Dr inż. Przemysław MAZUREK

Adiunkt w Katedrze Przetwarzania Sygnałów i Inżynierii Multimedialnej Zachodniopomorskiego Uniwersytetu Technologicznego w Szczecinie. Autor ponad 140 artykułów z zakresu cyfrowego przetwarzania sygnałów, estymacji ruchu, biopomiarów oraz przetwarzania obrazów biomedycznych.

e-mail: przemyslaw.mazurek@zut.edu.pl

Streszczenie

W artykule jest przedstawiona implementacja metody estymacji lakunarności z wykorzystanie przesuwnego okna dla GPGPU (programowalnych kart graficznych), umożliwiającą analizę wielorozdzielczą obrazu, w celu dalszej klasyfikacji. Porównano dwie implementacje – zwykłą oraz potokową (typu różnicowego) do przetwarzania obrazów w odcieniach szarości ze sterowanym progowaniem. Poprzez optymalizacje algorytmu obliczeniowego dla dużych rozmiarów okna analizy uzyskano 10-krotne przyspieszenie obliczeń.

Słowa kluczowe: Lakunarność, Analiza obrazów, Analiza wielorozdzielcza, GPGPU.

Estimation of lacunarity of a 2D object using GPGPU

Abstract

Multiresolution image analysis [1, 2] is important for pattern recognition applications. Wavelets and fractals [1, 2] are used typically. A fractal based technique for analysis of the placement of binary images using estimation of the lacunarity is possible. The lacunarity could be applied for the fractal and non-fractal objects (1D,2D,3D) [3]. The estimation of the lacunarity of a 2D object is based on the sliding windows approach. The number of pixels (1's) is counted (2) and the frequency table is computed (3). The normalization of this table gives the probability table (4). The lacunarity is calculated (7) using two moments obtained from this table. The different type of images (Fig.1) gives specific lacunarity plots (Fig.2), so classification is possible. The application of lacunarity to the grayscale images is also possible, e.g. using a set of thresholds. The computation of lacunarity is conceptually simple, but the implementation depends on processing platforms. Two implementations, conventional and pipeline, are compared in this paper. The conventional implementation uses counting of all pixels for the specific position of a window. The pipeline implementation supports the buffer of results so only updates are necessary. The programmable graphic card processor (GPGPU) and CUDA software platform are assumed for tests. The pipeline implementation is faster about 10 times for larger windows.

Keywords: lacunarity, image analysis, multiresolution analysis, GPGPU.

1. Wprowadzenie

Analiza wielorozdzielcza jest stosowana do analizy sygnałów jednowymiarowych i więcej wymiarowych [1]. Wykorzystuje ona analizę wybranym algorytmem ze zmiennym obszarem analizy. Ma to zastosowanie, gdy przykładowo obiekt (np. na obrazie) znajduje się blisko lub dalej od kamery. Oczekuje się, że te same cechy obiektu będą wyeksponowane w analizie wielorozdzielczej niezależnie od skali obiektu. W rzeczywistości można ją stosować do każdego rodzaju obiektów, w celu wyeksponowania określonych cech, które mogą być widoczne tylko dla określonej skali.

Analiza wielorozdzielcza wykorzystuje najróżniejsze algorytmy, najprostszym jest uśrednianie sygnału ze zmiennym rozmiarem okna. Szereg skal (okien analizy) może być dowolnie wybrany w zależności od aplikacji. Najbardziej kojarzone z analizą wielorozdzielczą są transformaty falkowe oraz analiza fraktalna [1, 2]. W przypadku falek, filtracja wykonywana jest z wykorzystaniem zestawu określonych wartości, wynikających z funkcji falki bazowej. Zadaniem falek jest analiza wielorozdzielcza bez utraty informacji oraz redundancji. Pozwala to na ponowną syntezę sygnału z uzyskanych wyników analizy. W przypadku wyłącznie analizy sygnału można, w zależności od potrzeb, zgodzić się na utratę informacji lub redundancję.

Analiza fraktalna związana jest ściśle z analizą wielorozdzielczą, gdzie zakłada się istnienie zależności między sygnałami we wszystkich skalach. Klasyczne podejście do analizy fraktalnej zakłada, że istnieje jedna liczba opisująca cechy obiektu niezależnie od skali, definiująca wymiar fraktalny, czyli stopień wypełnienia przestrzeni [2]. Analiza fraktalna może być stosowana do obiektów będących fraktalami, jednak może być także stosowana w pewnych przypadkach do obiektów niebędących fraktalami. Niektóre metody analizy pozwalają na to, inne nie.

Wynikiem analizy fraktalnej może być wymiar fraktalny jako jedna liczba, jednak jest to sytuacja w praktyce dosyć rzadka, ponieważ fraktale o jednej stałej wartości wymiaru są obiektami prawie zawsze fraktalami teoretycznymi. Zmiana wartości wymiaru fraktalnego w zależności od skali (okna analizy) jest typowa dla obiektów świata rzeczywistego. Najbardziej interesujące są przypadki, gdy analiza taka dopuszcza obiekty niebędące fraktalami. Daje to możliwości analizy różnych obiektów na potrzeby systemów rozpoznawania sygnałów, w szczególności klasyfikacji obrazów.

Jedną z metod analizy jest estymacja lakunarności. Może być ona stosowana do obiektów będących lub tez nie fraktalami, a wynikiem jest wykres zależny od skali.

2. Estymacja lakunarności

Estymacja lakunarności umożliwia analizę sygnałów 1D, 2D (obrazów), 3D (obiektów wolumetrycznych), Pierwotnie była stosowana do analizy sygnałów binarnych [3]. Estymacja lakunarności realizuje analizę grupowania oraz luk między wartościami. Założono, że wartość 1 należy do obiektu, a wartość 0 odpowiada luce.

Obrazy w odcieniach szarości mogą być analizowane za pomocą estymacji lakunarności bezpośrednio [3] lub pośrednio poprzez progowanie. Wartość progu może być dodatkowym parametrem analizy.

Skala określa rozmiar okna analizy, które rozmieszczane jest kafelkowo tworząc siatkę o stałym oczku lub stosuje się okno przesuwne. W przypadku okna przesuwnego wyniki są częściowo skorelowane, jednak występuje filtracja wyniku, z uwagi na zwiększoną ilość analizowanych przypadków tego samego obiektu, co jest korzystne.

Dla przebiegu 1D rozmiar okna jest zdefiniowany jako r, dla obrazu (2D) rozmiar boku okna jest pierwiastkiem kwadratowym z r (ilości pikseli w oknie). Rozmiar boku okna jest zmienny w procesie estymacji lakunarności i zakres zmian wynosi od 1 do połowy rozmiaru obiektu w danej osi układu współrzędnych [4]. Wartość maksymalna rozmiaru boku wynika z możliwości powstawania tzw. królików fatalnych (ang. fractal rabbits), które wynikają m.in. ze stosowania bardzo dużych skal i małej liczby przypadków, tworząc wartości wynikowe z szerokiego zakresu zmian [4, 5]. Zjawisko to jest typowe dla metod wielorozdzielczych.

Dla danego położenia i okna W wyznacza się ilość jedynek w oknie (np. pikseli czarnych o wartości 1):

$$s \in \langle 0, r \rangle \tag{1}$$

$$s_i = \sum_i W_i \tag{2}$$

Wartość dyskretna s_i jest wykorzystana do generacji tabeli częstości n, poprzez inkrementację:

$$n(s,r) \leftarrow n(s,r) + 1 \tag{3}$$

Tabela częstości służy do wyznaczenia tabeli prawdopodobieństw Q, z wykorzystaniem normalizacji:

$$Q(.,r) = \frac{n(.,r)}{\sum n(.,r)} \tag{4}$$

Wyznaczenie wartości lakunarności $\Lambda(r)$ wymaga wyznaczenia wartości pierwszego i drugiego momentu:

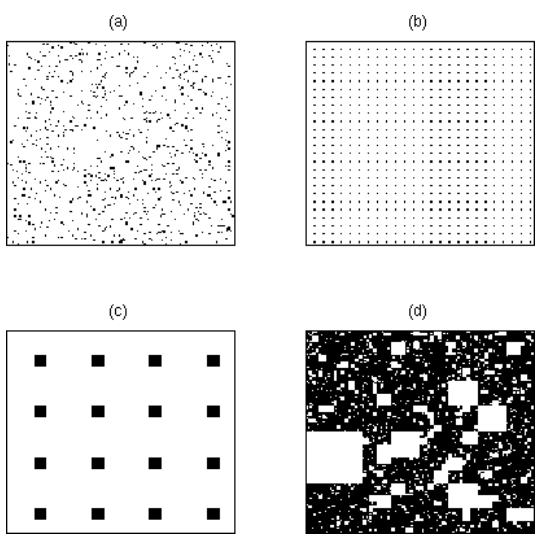
$$Z_1(r) = \sum_i s_i Q(s_i, r) \tag{5}$$

$$Z_2(r) = \sum_i s_i^2 Q(s_i, r) \tag{6}$$

$$\Lambda(r) = \frac{Z_2(r)}{[Z_1(r)]^2} \tag{7}$$

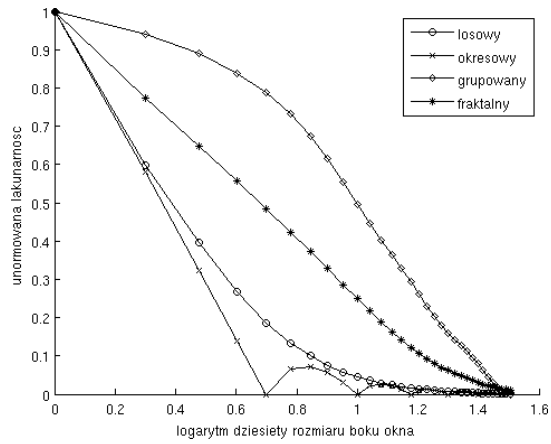
3. Przykładowe funkcje lakunarności

Wykres $\Lambda(r)$ zależy od ilości wartości 1. Gdy ich liczba jest taka sama dla różnej dyspersji jedynek, to dla $\Lambda(r=1)$ wartość jest taka sama. Na rys. 1 przedstawiono cztery przypadki konfiguracji dla obrazów: z losowym rozmieszczeniem pojedynczych jedynek (1), z równomierną gęstością jedynek (b), z grupowaniem jedynek w określonych obszarach (c) oraz z fraktalnym rozmieszczeniem jedynek (d). Wariant fraktalny jest dwuwymiarowym zbiorem Cantora z elementem losowym.



Rys. 1. Cztery przypadki rozmieszczenia jedynek (czarne piksele) na obrazach
Fig. 1. Example images of configuration of ones (black pixels)

Na rys. 2 przedstawiono wykres lakunarności dla przypadków z rys. 1. Różnice kształtu funkcji decydują o klasyfikacji typu przypadków. Przykładowo dla danych fraktalnych jest to linia prosta na wykresie logarytmicznym.



Rys. 2. Wykresu unormowanej lakunarności (log10) dla przypadków z rys. 1
Fig. 2. The normalized lacunarity (log10) for cases from Fig. 1

4. Implementacja estymacji lakunarności z wykorzystaniem GPGPU

Zasadniczy koszt obliczeniowy dla estymacji lakunarności jest związany z wyznaczeniem wartości w tabeli (3) częstości n. Operacja ta jest bardzo prosta (zliczanie ilości jedynek w danym obszarze przesuwnego okna), jednak zależy to od architektury. Najprostsze rozwiązanie sprzętowe dla struktur FPGA bazować może na licznikach, jednak w przypadku architektur o bardziej sztywnej konstrukcji konieczne jest stosowanie rozwiązań powiązanych z dostępną architekturą.

Układy GPGPU wykorzystują przetwarzanie równoległe o określonej organizacji, którą można częściowo zmieniać [6, 7]. W testowym rozwiązaniu wykorzystano nietypowe podejście z jednym blokiem w siatce. Blok ma organizację 512x1, co odpowiada ilości dostępnych wątków. Z uwagi na możliwość wykorzystania synchronicznego dostępu do pamięci zdecydowano się na organizację w postaci wiersza o długości podzielnej przez 32, co jest zalecane przez producenta. Dane (piksele) są reprezentowane jako liczby unsigned char. Nie jest to optymalne dla danych binarnych, jednak stosując zmienne wartości progu dla obrazu w 256 odcieniach szarości można uzyskać wykres lakunarności zmienny dodatkowo od wartości progu, co daje nowe możliwości analizy. Ilość i poziomy wartości progu mogą być definiowane w tym zakresie swobodnie.

Maksymalna szerokość obrazu to 512 pikseli, dla wykorzystanej karty, bez dodatkowych usprawnień implementacji. Wysokość jest dowolna, ale ograniczono się do 512 pikseli. Wątek przy uruchomieniu wartością otrzymuje długość boku kwadratu przesuwnego okna analizy. Piksele w tym oknie są progowane, a wynik akumulowany, po czym następuje uaktualnienie tabeli częstości. Każdy wątek operuje na indywidualnej tabeli częstości (pamięć współdzielona), z uwagi na brak możliwości wsparcia operacji atomowych przez użytą kartę. Dane obrazu są zgromadzone w pamięci globalnej karty graficznej.

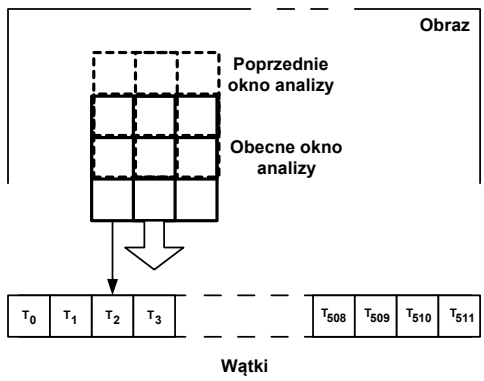
Wyniki końcowe w postaci 512 zestawów tabel częstości są przesyłane po zakończeniu obliczeń z pamięci współdzielonej do globalnej i dalej do CPU (host), gdzie są łączone i wyliczana jest lakunarność (4-7). Z uwagi na limit rozmiaru pamięci współdzielonej dla wybranej karty, możliwe jest obsłużenie 13 zakresów akumulacji (s) w tabeli. Zakresy te można definiować stosowanie do potrzeb lub wykonać dodatkowe przebiegi algorytmu w celu zwiększenia rozdzielczości (koszt jest liniowy).

5. Wyniki

Pojedynczy przebieg jest sterowany rozmiarem boku okna analizy oraz wartością progu. Ponieważ przetwarzanie z wieloma wartościami progu oraz z rozmiarami boku okna analizy wymaga

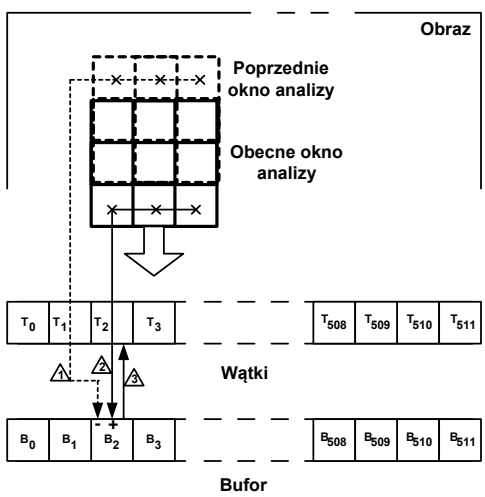
wielokrotnych uruchomień, to krytycznym staje się zwiększenie wydajności algorytmu.

Rozwiązanie zwykłe bazuje na zliczaniu wartości wewnątrz kwadratowego okna analizy (rys. 3). Nie jest to optymalne, ponieważ wyniki dla kolejnego położenia okna mogłyby być ponownie wykorzystane. Na rys. 5 przedstawiono linią kropkową osiągnięty rezultat dla karty Geforce 8600 GTS (procesor G82), taktowanie GPGPU: 1,45 GHz, taktowanie pamięci 1008 MHz, szyna danych 128-bit, dla platformy programowej CUDA 4.1. Porównawczo przedstawiono wyniki dla CPU (Pentium 4 2.66GHz).



Rys. 3. Zwykła implementacja
Fig. 3. The conventional implementation

Alternatywnym rozwiązaniem jest wariant potokowy w wariancie różnicowym, w którym tworzony jest potok w pionie (rys. 4). Wynik nowy bazuje na wyniku poprzednim, aktualizowanym przez odjęcie sumy ostatniego wiersza z poprzedniego położenia okna i dodanie sumy najnowszego wiersza z bieżącego położenia okna.



Rys. 4. Implementacja potokowa
Fig. 4. The pipeline implementation

Ponieważ współdzielenie wyników horyzontalnych między wątkami jest kłopotliwe, z uwagi małą pamięć współdzieloną oraz brak synchronizacji z wykorzystaniem instrukcji atomowych [6, 7] w ramach pamięci współdzielonej, zdecydowano się na każdorazowe wyliczanie sum poziomych dla każdego z wątku niezależnie. Dzięki synchronizmowi dostępu wątków w organizacji poziomej (kolejnych adresów pamięci globalnej) efekty kolizji z wielokrotnego dostępu są ograniczone.

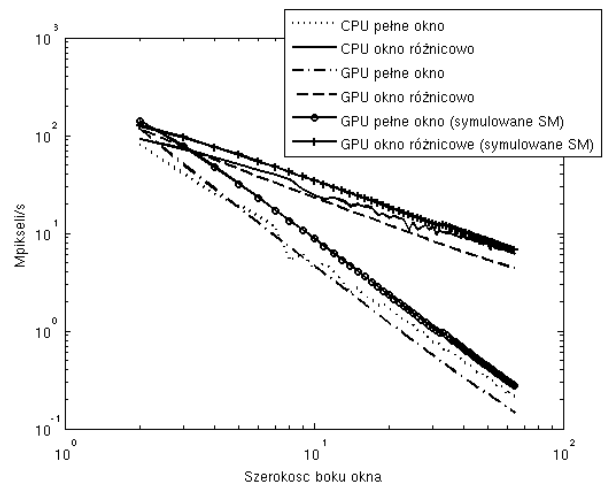
Wariant potokowy cechuje się wyższą wydajnością, dla dużych rozmiarów boku rzędu 10 razy, co jest istotne dla skrócenia czasu obliczeń (rys. 5).

Przy pomiarach wykorzystano pomiar najkrótszego czasu wykonania programu w części GPGPU wraz z transferem wyników do pamięci CPU dla 20 uruchomień. Wynika to z tego, że wpływ na pomiar ma system operacyjny, gdy karta GPGPU jest jednocześnie kartą graficzną podłączoną do monitora.

Czas przesyłu danych wejściowych (obraz) nie był brany pod uwagę, ponieważ zakłada się wielokrotne wykorzystanie tego samego bloku danych (różne progi, różne wartości rozmiaru boku okna analizy).

Symulowany dostęp do pamięci współdzielonej (SM) pozwala oszacować wydajność w przypadku dostępności obrazu z poziomu tylko tej pamięci, co pokazuje wpływ magistrali pamięci globalnej.

Wyniki dla CPU i GPGPU są zbliżone, przy czym wydajność CPU wynika z szybkiej pamięci podręcznej, w której przechowywany jest obraz. Wykorzystanie karty GPGPU o większej szerokości szyny danych oraz dużej pamięci współdzielonej może zmienić wynik na korzyść GPGPU.



Rys. 5. Porównanie wydajność przetwarzania implementacji
Fig. 5. Performance comparison of implementations

6. Wnioski

Pomiar lakunarności pozwala na analizę obrazu, zarówno binarnego, w odcieniach szarości oraz w odcieniach szarości z zestawem progów. Wykorzystanie tej techniki wymaga bardzo dużej ilości obliczeń, aczkolwiek stosunkowo prostych, to jednak wymagających odpowiedniej platformy przetwarzania oraz implementacji. Porównane dwie implementacje pokazują, możliwości implementacji z wykorzystaniem platformy CUDA 4.0. Możliwe jest rozszerzenie techniki potokowej dla analizy obiektów 3D.

Praca została wsparta z wykorzystaniem oprogramowania i sprzętu projektu UE EFRR ZPORR Z/2.32/I/1.3.1/267/05 "Badawczo-Dydaktyczne Centrum Nowoczesnych Technologii Multimedialnych Politechniki Szczecińskiej"

7. Literatura

[1] Peitgen H., Jurgens H., Saupe D.: Fractals for the Classrooms, Vol. 1, Springer-Verlag, 1991.
[2] Mandelbrot B.: The Fractal Geometry of the Nature, W.H. Freeman and Company, 1983.
[3] Plotnick R.E., Gardner R.H., Hargrove W.W., Prestegaard K., Perlmutter M.: Lacunarity analysis: A general technique for the analysis of spatial patterns, Physical Review E, 53 (5), 5461-5468, 1996.
[4] Kaye B.: A Random Walk Through Fractal Dimensions, VCH, 1994.
[5] Seuront L.: Fractals and Multifractals in Ecology and Aquatic Science, CRC Press, 2010.
[6] NVIDIA CUDA C Programming Guide v.4.0, NVidia, 2011.
[7] NVIDIA CUDA, CUDA C Best Practices Guide v.4.0, NVidia, 2011.

otrzymano / received: 09.05.2013
przyjęto do druku / accepted: 03.07.2013 *artykuł recenzowany / revised paper*