Mirosław SOCHA
AGH AKADEMIA GÓRNICZO-HUTNICZA, KATEDRA METROLOGII I ELEKTRONIKI,
Al. Mickiewicza 30, 30-059 Kraków

Szybka metoda estymacji położenia końcówki bronchofiberoskopu – implementacja w GPU

Mgr inż. Mirosław SOCHA

Mgr inż. Mirosław Socha ukończył Wydział Elektrotechniki, Automatyki, Informatyki i Elektroniki Akademii Górniczo-Hutniczej na specjalności Automatyka i Metrologia kierunku Elektrotechnika. Pracuje na stanowisku asystenta w Katedrze Metrologii i Elektroniki AGH. Zajmuje się dydaktyką w zakresie wizualizacji i pomiarów wielkości nieelektrycznych oraz problemami wizualizacji i przetwarzania danych medycznych.

e-mail: socha@agh.edu.pl

Streszczenie

W pracy przedstawiono szybką metodę szacowania położenia końcówki bronchofiberoskopu, będącej ważnym fragmentem prototypowego systemu do wspomagania zabiegów bronchoskopowych. Omówiono sposób działania algorytmu śledzenia ruchu endoskopu oraz przedstawiono szczegóły nowej implementacji algorytmu, która wykorzystuje możliwości obliczeniowe procesorów kart graficznych. Zastosowanie wielordzeniowych procesorów graficznych GPU do przetwarzania obrazów z endoskopu zaowocowało ponad 25. krotym przyśpieszeniem algorytmu.

Słowa kluczowe: przetwarzanie obrazów, nawigacja, bronchoskopia, wspomaganie zabiegów, akceleracja obliczeń, GPGPU.

Fast estimation of bronchofiberoskope ego-motion – GPU based implementation

Abstract

In this work a new implementation of fast approximation of bronchofiberoscopy ego-motion is presented. This algorithm is an important part of a prototype system to support bronchofiberoscopic treatment. Its goal is to help a doctor to take a sample of a pathological lesion (found in computed tomography scan) by means of needle aspiration, performed from a bronchial tree interior [3, 4]. The approach is based on real-time registration of the 2D endoscopic images and virtual ones generated by a virtual camera located inside a 3D CT-based model of the bronchial tree. To speed up ego-motion estimation [5] in bronchial environment there is used a simplified model of geometric relations based on the cylindrical shape accompanied by the fixation on a carina [6], which reduces the number of degrees of freedom of the motion to four. It is achieved by continuous tracking of the carina (stationary point) illuminated by the camera light source, and by analyzing bronchial wall radial moves relative to the fixed point by correlation in the polar coordinates. Fig. 1 shows estimation of rotation steps, Fig. 2 estimation of translation. Use of the multi-core graphics processing unit (GPU) to process the images from the endoscope allowed reducing the computation time more than 25 times.

Keywords: image processing, navigation, bronchoscopy, GPGPU.

1. Wprowadzenie

Bronchofiberoskopia jest małoinwazyjnym badaniem pozwalającym wzrokowo ocenić stan dolnych dróg oddechowych człowieka. Oceny tej dokonuje lekarz bronchoskopista poprzez umieszczenie bronchofiberoskopu w drzewie oskrzelowym pacjenta i analizę obrazu dostarczanego przez bronchofiberoskop. Jednym z zabiegów współczesnej diagnostyki zmian chorobowych klatki piersiowej, wykonywanych podczas bronchofiberoskopii jest przezoskrzelowa biopsja aspiracyjna. Zabieg ten polega na pobraniu próbek tkanek, zazwyczaj z powiększonych węzłów chłonnych, znajdujących się za ścianą drzewa oskrzelowego. Pobrane próbki są następnie poddawane analizie histopatologicznej w celu stwierdzenia obecności tkanek nowotworowych. Pobranie aspiratu z węzłów chłonnych jest zadaniem trudnym, ponieważ lekarz bronchoskopista wybierając punkt nakłucia nie widzi węzłów chłonnych na obrazie z endoskopu. Wybór miejsca biopsji dokonywany jest na podstawie danych z tomografii komputerowej (serii obrazów dwuwymiarowych), własnej wyobraźni przestrzennej oraz doświadczenia. Jedną z coraz częściej stosowanych metod wspomagania zabiegów transbronchialnej biopsji aspiracyjnej jest ultrasonografia wewnątrzoskrzelowa wykonywana w czasie rzeczywistym (ang. *real-time endobronchial ultrasound guided transbronchial needle aspiration*, EBUS-TBNA) [1]. Metoda ta wymaga jednak zastosowania specjalistycznego endoskopu wyposażonego w sondę ultrasonograficzną. Opisana w niniejszej pracy metoda śledzenia końcówki endoskopu może stanowić uzupełnienie metody EBUS-TBNA lub może ją zastępować, zwłaszcza w ośrodkach, które nie dysponują odpowiednim sprzętem [2]. Komputerowe systemy planowania i wspomagania zabiegów bronchoskopowych umożliwiają pełniejsze wykorzystanie danych z tomografii komputerowej, wykonywanej zazwyczaj przed bronchoskopią oraz obrazów pochodzących z kamery bronchofiberoskopu [2-4]. Za prekursora techniki wspomagania zabiegu bronchoskopii, poprzez analizowanie obrazu rzeczywistego z bronchoskopu, uznawany jest Mori [7]. Rozwinięciem jego prac jest prototyp systemu wspomagania zabiegu bronchofiberoskopii opracowany w Katedrze Metrologii i Elektroniki AGH [4, 8, 9]. System ten został przystosowany głównie do wspomagania biopsji wykonywanych w tchawicy.

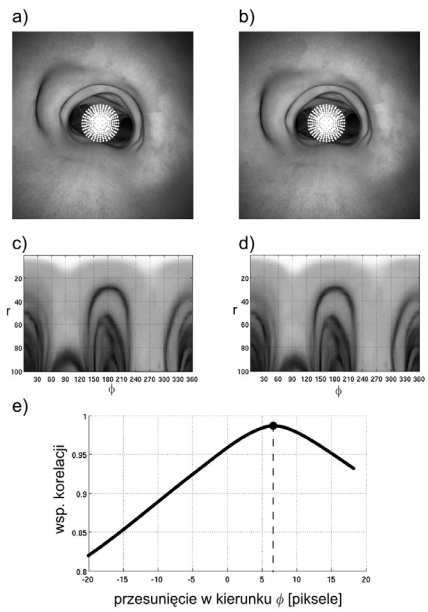
Opracowany system wspomagania zabiegu opiera się na dwóch źródłach informacji: danych z tomografii komputerowej oraz obrazie z kamery endoskopu. Na podstawie danych z tomografii komputerowej rekonstruowana jest przestrzenna powierzchnia drzewa oskrzelowego pacjenta. Następnie, tak otrzymany obraz wirtualnego drzewa oskrzelowego pacjenta porównywany jest z obrazem rzeczywistym. Zmieniając pozycję wirtualnej kamery szukana jest taka pozycja w przestrzeni, przy której obraz wirtualny i rzeczywisty są do siebie najbardziej podobne. Zastosowaną miarą podobieństwa jest informacja wzajemna [8]. W celu ograniczenia liczby iteracji algorytmu dopasowywania obrazów, opracowano szybką metodę szacowania ruchu końcówki bronchofiberoskopu [9].

2. Metoda szacowania położenia endoskopu

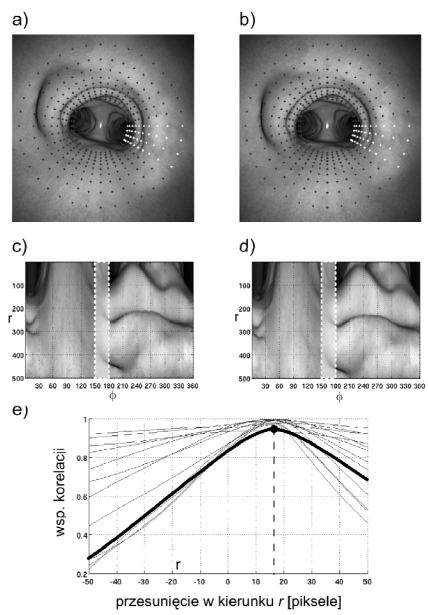
Opracowany algorytm estymacji ruchu końcówki bronchofiberoskopu bazuje na analizie obrazów z endoskopu [9]. Poprzez korelowanie obrazów ścian, po wcześniejszym usunięciu zniekształceń wprowadzanych przez optykę endoskopu, możliwe jest szybkie i dokładne wyznaczanie zarówno przesunięć końcówki endoskopu w głąb drzewa oskrzelowego, jak i wyznaczenie jego rotacji. Duża wydajność obliczeniowa algorytmu jest konsekwencją przyjętego, uproszczonego, modelu segmentów drzewa oskrzelowego w postaci cylindra. Dodatkowym założeniem jest śledzenie rozbłysku światła na ostrodze [6]. Rozbłysk ten traktowany jest jako punkt odniesienia, na który patrzy kamera. Umożliwia on ograniczenie do czterech stopni swobody analizę ruchu kamery. Ruch postępowy oraz obroty estymowane są na podstawie korelacji obrazu ścian, obliczanej w cylindrycznym układzie współrzędnych.

Na rys. 1 przedstawiono obraz wirtualnego drzewa oskrzelowego, wygenerowany w dwóch chwilach czasowych (rys. 1a i 1b). Wirtualna kamera w tym czasie została przesunięta oraz obrócona. Na obrazach tych, w pierwszej kolejności wykryto rozbłysk pochodzący od oświetlenia endoskopu. Następnie, traktując rozbłysk jako środek biegunowego układu współrzędnych, dokonano „rozwinięcia" otoczenia rozbłysku (rys. 1c i 1d). W celu oszacowania

kąta obrotu, tak otrzymane wycinki obrazu są korelowane w kierunku osi ϕ.



Rys. 1. Estymacja obrotu endoskopu: a-b) obraz tchawicy w chwili t_1 i t_2, c-d) otoczenie rozbłysku rozwinięte do układu biegunowego, e) wartości współczynników korelacji wzajemnej obliczonych dla obrazów
Fig. 1. Estimation of rotation: a-b) frame in time t_1 and t_2, c-d) interpolated data around flare, e) croscorrelation coefficients in ϕ direction



Rys. 2. Estymacja obrotu endoskopu: a-b) obraz w chwili t_1 i t_2, c) ściany drzewa oskrzelowego rozwinięte do układu biegunowego dla chwili t_1 i t_2, e) wartości obliczonych w kierunku r współczynników korelacji
Fig. 2. Estimation of translation: a-b) frame in time t_1 and t_2, c-d) unrolled wall of trachea, e) image croscorrelation coefficients in r direction

Estymacja przesunięcia jest realizowana w podobny sposób – odbywa się również przez analizę wycinków obrazów, jednak w tym przypadku korelowane są fragmenty rozwiniętych ścian drzewa oskrzelowego. Na rys. 2a i 2b ciemnymi kropkami zaznaczono obszar, który jest przekształcany do układu biegunowego. Dodatkowo jasnymi punktami zaznaczono jeden z korelowanych fragmentów ścian drzewa oskrzelowego. W przypadku estymacji ruchu nie są korelowane całe „rozwinięte" ściany, ale ich fragmenty. Dzięki temu, możliwe jest śledzenie ruchu postępowego w wielu kierunkach. Dla każdego fragmentu obrazu ścian (obszar jaśniejszy na rys. 2c i 2d) wyznaczane są współczynniki korelacji w kierunku osi r biegunowego układu współrzędnych (rys. 2e).

Wartość kąta obrotu końcówki endoskopu obliczana jest na podstawie pozycji wartości maksymalnej (rys. 1e) zaś przesunięcie obliczane jest jako wartości średniej geometrycznej wektorów cząstkowych (rys. 2e).

3. Architektura procesora graficznego GPU

Współczesne karty graficzne wyposażone są w specjalizowane, wielordzeniowe procesory graficzne (ang. *Graphics Processing Unit* – GPU), których stopień skomplikowania oraz wydajność są znacznie większe od najwydajniejszych procesorów ogólnego zastosowania (ang. *Central Processing Unit* – CPU). Teoretyczna, maksymalna wydajność procesorów CPU jest na poziomie 500 GFLOP/s (milionów operacji zmiennoprzecinkowych na sekundę), zaś procesorów graficznych GPU przekracza 3000 GFLOP/s. Opracowana przez firmę NVIDIA zunifikowana architektura procesora graficznego CUDA™ (ang. *Compute Unified Device Architecture*) umożliwia wykonywanie dowolnych obliczeń przez procesor karty graficznej [10].

Bardzo wysoka wydajność GPU została osiągnięta przez zastosowanie odmiennej od CPU architektury. Podstawową różnicą między procesorem GPU a CPU jest ograniczenie w GPU do minimum części kontrolnej i pamięci podręcznej oraz umieszczenie w jednym procesorze GPU bardzo dużej liczby rdzeni wykonujących obliczenia arytmetyczno-logiczne, stało- oraz zmiennoprzecinkowe. Rdzenie GPU nie mają pełnej funkcjonalności rdzenia procesora CPU – są wyspecjalizowane do równoczesnego przetwarzania danych przez stosunkowo krótkie programy. Rdzenie GPU wykonują równolegle ten sam program (ang. *kernel*) na dużym zbiorze danych – realizują przetwarzanie równoległe na poziomie danych (ang. *Single-Program Multiple-Data* — SPMD). Najnowsze procesory GPU z rodziny Fermi i Kepler umożliwiają tworzenie wątków. Dzięki temu możliwe jest wykonywanie wielu programów w tym samym czasie (ang. *concurrent kernel execution*). Pojedynczy procesor GPU składa się z dużej liczby rdzeni, która może się wahać od kilkunastu, w procesorach niskobudżetowych, do kilku tysięcy (np. 2688 w procesorze karty GeForce GTX TITAN), w najwydajniejszych wersjach przeznaczonych na rynek konsumencki.

4. Implementacja algorytmu szacowania położenia w GPU

Użycie procesora graficznego GPU (ang. *Graphics Processing Unit*) do wspomagania obliczeń jest obecnie szczególnie uzasadnione, biorąc pod uwagę powszechność występowania tych procesorów w nowoczesnych komputerach klasy PC oraz dostępność narzędzi programistycznych, ułatwiających wykonywanie dowolnych obliczeń z użyciem GPU.

Z przeprowadzonych testów wydajności algorytmu estymacji ruchu endoskopu wynikało, że mimo zastosowania znacznych uproszczeń w modelu otoczenia i ruchu endoskopu, algorytm estymacji był na tyle skomplikowany, że nie mógł przetwarzać danych w czasie rzeczywistym (między kolejnymi ramkami obrazu) [4]. W technologii CUDA, wykorzystującej moc obliczeniową procesorów GPU, zaimplementowano dwa kluczowe pod względem czasu obliczeń fragmenty kodu: interpolację danych w celu przygotowania ich do dalszego przetwarzania oraz obliczanie unormowanej funkcji korelacji.

Podstawowym założeniem podczas implementacji było przeniesienie wszystkich danych do szybkiej pamięci karty graficznej, a następnie równoległe przetwarzanie ich z wykorzystaniem setek procesorów. W przypadku interpolacji danych, jeden piksel obrazu wynikowego obliczany jest przez jeden rdzeń GPU.

4.1. Przygotowanie obrazu do analizy

Przygotowanie danych do korelacji składa się z następujących etapów:

- przesłania kolorowego obrazu z endoskopu do pamięci karty graficznej,
- korekcji nieliniowych zniekształceń optycznych, wprowadzanych przez szerokokątną optykę endoskopu [4],
- konwersji obrazu z kolorowego do odcieni szarości,
- wyznaczenia pozycji rozbłysku, najjaśniejszego punktu znajdującego się w centrum obrazu,
- interpolacji otoczenia rozbłysku – przygotowania danych do estymacji obrotu,
- rozwinięcia ścian drzewa tchawicy – przygotowania danych do estymacji przesunięcia.

Procesor GPU zawiera wyspecjalizowane jednostki do sprzętowego adresowania i filtrowania tekstur [10]. Wykorzystanie tych jednostek umożliwia wydajne interpolowanie obrazów. Jednostki te mogą operować jedynie na specjalnym obszarze pamięci jakim jest pamięć tekstur (ang. *texture memory)*. W pamięci tej umieszczane były obrazy, które wymagały adresowania subpikselowego: oryginalny obraz z endoskopu oraz obraz po korekcji zniekształceń i usunięciu koloru.

Korekcja zniekształceń obliczana była dla każdej klatki filmu pochodzącego z endoskopu. Zależność między współrzędnymi pikseli obrazu źródłowego i skorygowanego opisuje równanie wielomianowe [4]. Ponieważ przekształcenie to jest stałe dla konkretnego bronchoskopu, odpowiadające sobie współrzędne pikseli zostały stabelaryzowane. W tym celu utworzono dwuwymiarową tablicę o wymiarach obrazu po korekcji, w której umieszczono obliczone współrzędne pikseli obrazu nieskorygowanego.

W podobny sposób przygotowano tablice współrzędnych pikseli do estymacji kąta rotacji (rozwinięcie okolic rozbłysku do polarnego układu współrzędnych) oraz przesunięcia (rozwinięcie ścian drzewa oskrzelowego na cylinder). W obu przypadkach rozwinięcie odbywa się wokół punktu fiksacji, którego współrzędne wyznaczane są dla każdej klatki na podstawie pozycji rozbłysku. Wartości stabelaryzowane wyznaczono dla punktu rozwinięcia (początku biegunowego układu współrzędnych) o współrzędnych (0,0). W trakcie tworzenia obrazów do korelacji, współrzędne zapisane w tablicach są przesuwane o wartość równą współrzędnym bieżącego punktu fiksacji.

Program korygujący zniekształcenia wywoływany jest dla każdego piksela – rozmiar tablicy rdzeni (tzw. *grid*) jest równy rozmiarom obrazu po korekcji. Każdy rdzeń GPU, do którego trafia program, oblicza na podstawie zmiennych wbudowanych współrzędne (x_{out},y_{out}) „swojego" piksela. Współrzędne te służą do zaadresowania tablicy ze stabelaryzowanymi współrzędnymi pikseli (x_{in},y_{in}) obrazu źródłowego. W kolejnym kroku program, z wykorzystaniem funkcji sprzętowego adresowania tekstur *tex2D*, pobiera z obrazu źródłowego wartość koloru dla zadanych współrzędnych. Jeżeli zadane współrzędne nie są liczbami całkowitymi (nie „trafiają" w piksel), wykonywana jest sprzętowa dwuliniowa interpolacja danych. Ostatnim etapem algorytmu jest konwersja kolorowego obrazu do skali szarości poprzez obliczenie luminancji. Wynik zapisywany jest w tablicy znajdującej się w obszarze pamięci globalnej (ang. *global memory)*, która zapewnia swobodny dostęp do danych.

Wyznaczenie współrzędnych rozbłysku odbywa się dwuetapowo. W pierwszej fazie monochromatyczny obraz pozbawiony zniekształceń wymnażany jest z wcześniej przygotowanym i stabelaryzowanym dwuwymiarowym oknem Gaussa. Wartość maksymalna okna znajduje się w środku macierzy. Rozmiar tablicy zawierającej współczynniki okna jest dwa razy większy od rozmiaru obrazu po korekcji. Uprościło to wyznaczanie współrzędnych środka okna i rozwiązało problem sprawdzania zakresów tablic przy wymnażaniu współczynników okna z wartościami obrazu. Dodatkowo, możliwe jest wymnożenie okna z obrazem przy dowolnym przesunięciu. Drugim krokiem było wyznaczenie współrzędnych rozbłysku występującego na rozwidleniu głównym tchawicy. Wydajna implementacja wyszukiwania wartości maksymalnej w środowisku wielordzeniowy nie jest zadaniem trywialnym, ponieważ wymaga przeglądnięcia i uwzględnienia wartości z wszystkich analizowanych danych. Operacja taka nie może być zrealizowana jednocześnie przez wiele pracujących równolegle procesorów. Problem ten nosi nazwę *równoległej redukcji*, a jego rozwiązanie zostało szczegółowo przedstawione na przykładzie sumowania w pracy [11].

Sekwencyjne przeglądanie tablicy w poszukiwaniu wartości największej jest możliwe, ale zupełnie nieefektywne obliczeniowo. Wersja realizująca wyszukiwanie wartości maksymalnej opiera się na kaskadowym wywoływaniu programów: każdy rdzeń dokonuje porównania małego fragmentu danych, a wynik zapisywany jest do szybkiej pamięci wspólnej (ang. *shared memory*), tworząc dane wejściowe dla kolejnego uruchomienia programu. W kolejnych iteracjach algorytmu liczba porównywanych danych jest zmniejszana, aż ostatecznie otrzymywana jest jedna wartość wynikowa. Jak pokazują przykłady zaprezentowane w pracy [11], uwzględnienie architektury procesora GPU oraz optymalizacja kodu umożliwiają nawet trzydziestokrotne skrócenie czasu obliczeń. Opracowana przez autora implementacja wyszukiwania współrzędnych rozbłysku bazuje na najszybszym przykładzie zaprezentowanym w pracy [11]. Przykład ten został zmodyfikowany w taki sposób, by w wyniku obliczeń uzyskiwane były współrzędne rozbłysku.

Po skorygowaniu zniekształceń, konwersji barw i wyznaczeniu punktu fiksacji (rozbłysku) uruchamiane są dwa programy, których zadaniem jest „rozwinięcie" odpowiednich fragmentów obrazu do układu biegunowego oraz na powierzchnię cylindra. Tak jak przy korygowaniu zniekształceń, również tutaj obraz źródłowy (skorygowany i monochromatyczny) kopiowany jest z pamięci współdzielonej do pamięci tekstur. Dzięki temu, możliwe jest użycie funkcji *tex2D* do próbkowania ze sprzętową interpolacją pikseli obrazu źródłowego. Analogicznie do programu korekcji zniekształceń, współrzędne próbkowanych pikseli są odczytywane z wcześniej przygotowanej tablicy. W kolejnym kroku tak otrzymane wartości są korygowane o przesunięcie wynikające z pozycji rozbłysku.

W wyniku działania programów dla jednej ramki obrazu wyznaczane są dwie macierze o zadanych rozmiarach, w których znajdują się „rozwinięte" okolice rozbłysku (rys. 1c) oraz ściany tchawicy (rys. 1d). Ponieważ obydwa zbiory danych korelowane są jednowymiarowo, ale w dwóch różnych kierunkach, macierz do wyznaczania przesunięcia jest transponowana już na poziomie tworzenia. Ma to na celu uproszczenie implementacji funkcji korelującej.

4.2. Korelowanie obrazów

Przygotowane w pamięci karty graficznej tablice, zawierające przetransformowane fragmenty obrazów wejściowych (I oraz T), są korelowane dla kolejnych przesunięć (u,v) z wykorzystaniem metody unormowanej korelacji wzajemnej NCC [12]:

$$K_{I,T}(u,v)=\frac{\sum_{x,y}\left(I(x,y)-\overline{I}\right)\cdot\left(T(x-u,y-v)-\overline{T}\right)}{\sqrt{\sum_{x,y}\left(I(x,y)-\overline{I}\right)^2\cdot\sum_{x,y}\left(T(x-u,y-v)-\overline{T}\right)^2}} \quad (1)$$

Zaimplementowane obliczenia NCC prowadzone są w trzech etapach:

1. w pierwszym kroku z obrazów usuwana jest wartość średnia: suma obrazów wyznaczana jest kaskadowo (równolegle, przez wiele procesorów [11]) i dzielona przez rozmiar korelowanych danych,

2. w kolejnym kroku obliczany jest czynnik normalizujący: wartości pikseli są podnoszone do kwadratu, a następnie

sumowane kaskadowo, tak uzyskane sumy obrazów są wymnażane i pierwiastkowane,

3. następnie wyznaczana jest wartość korelacji dla zadanego zbioru przesunięć w jednej osi, dla każdego przesunięcia u wykonywane są następujące obliczenia:
   a. dla każdego piksela fragmentu korelowanego obrazu uruchamiany jest program CUDA, którego zadaniem jest wymnożenie obrazu o współrzędnych $I_1(x,y)$ z obrazem $I_2(x-u,y)$, w wyniku równoległego mnożenia obrazów powstaje macierz o rozmiarze takim samym jak korelowane obrazy,
   b. otrzymana macierz zostaje zsumowana (licznik równania 1),
   c. wynik sumowania jest normowany wcześniej obliczoną wartością i zapisywany do macierzy wyników, pod indeksem u.

W kolejnych iteracjach programu obliczenia wykonywane na najnowszym obrazie są zapisywane do ponownego użycia.

Na podstawie analizy zapisów wideo zawierających nagrania zabiegów bronchoskopowych można stwierdzić, że ruchy wykonywane endoskopem w trakcie „przeglądania" drzewa oskrzelowego przez lekarza, charakteryzują się niewielkimi prędkościami przesuwu i obrotu. Fakt ten został wykorzystany do ograniczenia liczby wykonywanych korelacji – współczynniki korelacji wyliczane są jedynie dla wąskiego zakresu przesunięć. Dla macierzy przesunięć arbitralnie przyjętym zakresem było $u \in$[-50,50], zaś w przypadku obrotu $u \in$[-20,20].

Dane do korelowania umieszczane są w pamięci tekstur, do której odwołania są buforowane w pamięci podręcznej (ang. *cache*). Dodatkowo możliwe jest utworzenie z przesłanych danych bufora cyrkulacyjnego – odwołania do pikseli o indeksach x większych niż rozmiary obrazu N_x są zamieniane bądź na wartość piksela „na krawędzi" obrazu $x'=N_x$, bądź na wartość równą $x'=mod(x,N_x)$. Adresowanie *modulo* zostało wykorzystane przy korelowaniu fragmentów obrazów przy estymacji obrotu. Dane te „zapętlają się" w wierszach, dzięki temu podczas wyliczania współczynnika korelacji zawsze wymnażane są wszystkie elementy macierzy. Dzięki temu, współczynnik normalizujący może być obliczony jednokrotnie dla wszystkich przesunięć u.

W przypadku estymacji przesunięcia, wyniki interpolacji danych (rozwinięcie ścian tchawicy na cylinder) są najpierw transponowane, potem dzielone na mniejsze fragmenty, a następnie korelowane. Ponieważ w tym przypadku dane w wierszach nie są „zapętlone" (kolejne piksele wiersza znajdowały się coraz głębiej tchawicy), niezbędne jest ograniczenie rozmiarów korelowanych danych w taki sposób, by korelowany piksel „nie wyszedł" poza rozmiar wycinka obrazu. W tym celu algorytm koreluje jedynie $N_x-2 \cdot u$ pikseli w każdym wierszu.

Zgodnie z definicją unormowanej korelacji wzajemnej obrazów, normalizacja powinna być przeprowadzana dla każdego przesunięcia (u,v) (równanie 1). W celu uproszczenia obliczeń przyjęto stałą wartość współczynnika normalizacji, wyznaczanego jednokrotnie dla każdego wydzielonego do korelacji fragmentu. Uproszczenie to nie ma wpływu na działanie algorytmu, ponieważ nie jest szukana wartość maksymalna korelacji, ale jej położenie w macierzy współczynników korelacji. Dodatkowo można zauważyć, że rozpatrywane przesunięcia są relatywnie niewielkie w porównaniu z rozmiarem korelowanego wycinka.

5. Wnioski

Wyniki pomiarów czasu wykonywania poszczególnych fragmentów algorytmu szacowania ruchu endoskopu zebrano w tabeli 1. Wszystkie eksperymenty przeprowadzono dla tego samego zbioru stu obrazów testowych pochodzących z bronchofiberoskopu. Wyniki pomiarów zostały uśrednione.

Zgodnie z przypuszczeniami, w wyniku równoległego przetwarzania niewielkiej porcji danych, czasy wykonania poszczególnych fragmentów kodu algorytmu szacowania przesunięcia końcówki endoskopu uległy znacznemu skróceniu.

Tab. 1. Wyniki pomiarów otrzymane z wykorzystaniem procesora Core2Quade 2.4 GHz oraz karty graficznej GeForce 260
Tab. 1. The results of measurements of parts algorithm computation time obtained from the use of Core2Quade 2.4 GHz and graphics card GeForce 260

Fragment algorytmu	Czas obliczeń [ms]		
	Matlab	C++	CUDA
Korekcja nieliniowości obrazu	52	18	0,06
Wykrycie rozbłysku	10	13	0,02
Przygotowanie danych (interpolacja)	74	23	0,04
Estymacja ruchu endoskopu (korelacja)	692	242	9,12

Uzyskane przyśpieszenie działania algorytmu umożliwia szacowanie przesunięcia i obrotu kamery endoskopu w czasie krótszym niż 20ms, czyli przed nadejściem kolejnej ramki obrazu. Czas ten może być skrócony poprzez ograniczenie rozmiarów korelowanych macierzy. Jednocześnie użycie nowocześniejszej karty graficznej, o większej liczbie procesorów działających wydajniej, gwarantuje dalsze skrócenie czasu obliczeń.

6. Literatura

[1] Herth F.J., Eberhardt R., Vilmann P., Krasnik M., Ernst A.: Real-time endobronchial ultrasound guided transbronchial needle aspiration for sampling mediastinal lymph nodes. Thorax; vol. 61: str. 795–798, 2006.

[2] Duplaga M., Socha M., Wojciechowski W., Tomaszewska R., Soja J., Urbanik A., Sładek K., Niżankowska-Mogilnicka E.: Porównanie skuteczności wirtualnej bronchoskopii i ultrasonografii wewnątrzoskrzelowej jako metod wspomagających transbronchialną aspiracyjną biopsję igłową u pacjentów z podejrzeniem raka płuc, Przegląd Lekarski, t. 64, str. 36-41, 2007.

[3] Bartz D.: Virtual Endoscopy in Research and Clinical Practice; Eurographics – Computer Graphics forum, vol. 24, nr 1, str. 111-126, 2005.

[4] Bulat J., Duda K., Socha M., Turcza P., Zielinski T. P., Duplaga M.: Computational tasks in computer-assisted transbronchial biopsy. Future Generation Comp. Syst. (FGCS) 26(3):455-461, 2010.

[5] Bernard C.: Discrete Wavelet Analysis for Fast Optic Flow Computation; Applied and Computational Harmonic Analysis, vol. 11, nr 1, str. 32-63, 2001.

[6] Daniilidis K.: Fixation simplifies 3D Motion Estimation, Computer Vision and Image Understanding, vol. 68 (1997), nr 2, str. 158-169

[7] Mori K., Deguchi D., Sugiyama J., et al.: Tracking of a Bron-choscope Using Epipolar Geometry Analysis and Intensity-Based Image Registration of Real and Virtual Endoscopic Images; Medical Image Analysis, vol. 6, nr 3, str. 321-336, 2002.

[8] Turcza P.: Navigation system for bronchofiberoscopic procedures based on image registration with scale adaptive image similarity measure; European Signal Processing Conference EUSIPCO-2005, Antalya (Turcja), 2005.

[9] Twardowski T., Zieliński T., Duda K., Socha M., Duplaga M.: Fast estimation of broncho-fiberoscope egomotion for CT-guided transbronchial biopsy; IEEE Int. Conference on Image Processing ICIP-2006, Atlanta, str. 1189-1192, 2006.

[10] NVIDIA: CUDA. C Programming Guide. nVidia Corp, http://docs.nvidia.com/cuda 2013.

[11] Harris, M.: Optimizing Parallel Reduction in CUDA. nVidia. 2008.

[12] Lewis, J. P.: Fast normalized cross-correlation. Industrial Light & Magic. 1995.

otrzymano / received: 18.12.2012
przyjęto do druku / accepted: 01.02.2013 *artykuł recenzowany / revised paper*