Paweł Forczmański, Przemysław Szeptycki

INDEKSOWANIE DUŻYCH ZBIORÓW OBRAZÓW

Streszczenie

W artykule zaprezentowane zostały wybrane metody indeksowania dużych zbiorów obrazów statycznych bazujące na autorskich dokonaniach i na deskryptorach zdefiniowanych przez standard MPEG-7 (ISO/IEC 15938). Omawiane deskryptory opisują cechy niskopoziomowe związane przede wszystkim z kolorem i jego rozkładem na obrazie oraz cechy teksturalne. Przeprowadzone badania doprowadziły do wyłonienia najbardziej użytecznych deskryptorów w zadaniach przeszukiwania graficznych baz danych i tworzenia foto-mozaik. Wyniki pracy zostały zaimplementowane w formie programów komputerowych.

Słowa kluczowe:

Indeksowanie, kolor, tekstura, histogram, MPEG-7,CBIR

1. Wstęp

Poprzez indeksowanie dowolnego zbioru danych rozumie się proces tworzenia i utrzymywania indeksu umożliwiającego obniżenie czasu dostępu do danych. Indeksem jest w tym przypadku obiekt o dużo mniejszej złożoności (wielkości) reprezentujący jednoznacznie element zbioru. Dostęp do danych jest realizowany poprzez indeksy na zasadzie: zapytanie →indeks1→wyszukiwanie→indeks2→element(y) zbioru.

Indeksy, w zależności od zastosowania, są tworzone wprost z danych, np. poprzez wybór ich fragmentów, przy użyciu matematycznych funkcji skrótu (MD5, SHA1) lub też za pomocą różnorakich transformat i redukcji wymiarowości (DCT [13], falki [12], KLT/PCA [3]). Często indeks zawiera informacje, które nie są wprost reprezentowane w danych. Taki przypadek obejmuje zadania indeksowania zbiorów danych multimedialnych, które stanowią obszerne i złożone struktury. Dla danych wizualnych (obrazy statyczne, sekwencje video) stosuje się dwa rodzaje indeksów:

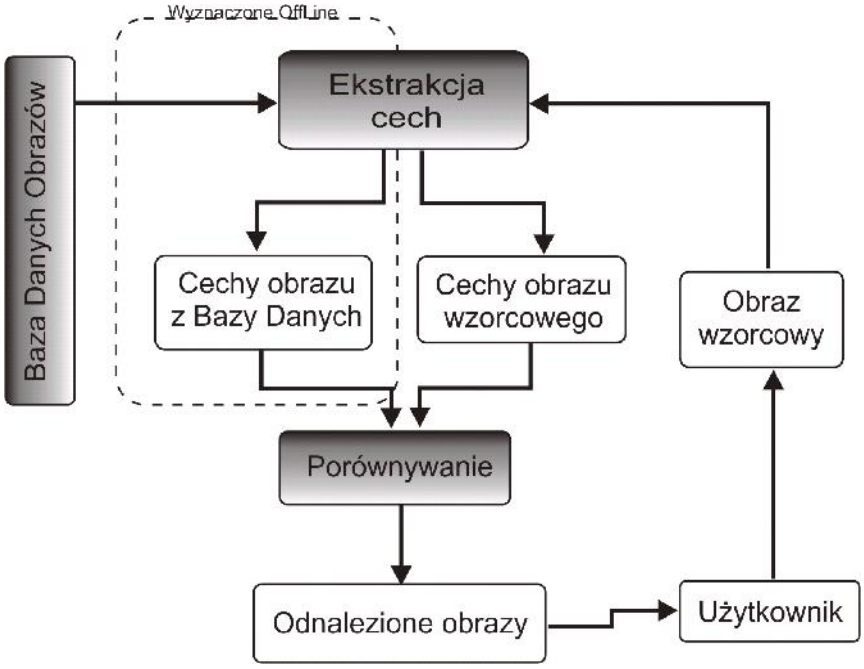
- wysokopoziomowe (opisowe), które są nadawane arbitralnie w sposób manualny i mają charakter hasłowy dotyczący zawartości, autorstwa, czasu utworzenia i przynależności do odpowiedniej klasy danych,
- niskopoziomowe, które reprezentują informację o cechach fizycznych i percepcyjnych, np. kolor, kształt, wzór (teksturę), ruch, aktywność itp.

Indeksy pierwszego rodzaju stosowane są w tradycyjnych systemach bazodanowych i wyszukiwarkach internetowych. Indeksy drugiego rodzaju są wykorzystywane obecnie dość wąsko, głownie w systemach gromadzących materiały video i dużych bazach danych obrazów statycznych. Większość z rozwiązań pozostaje niestety tylko na etapie zaawansowanych prac badawczych [7,9,11] i nie jest dostępna szerokiemu gronu użytkowników. Znaczący krok w kierunku stworzenia w pełni automatycznych systemów indeksujących został poczyniony wraz z nadejściem standardu MPEG-7, który definiuje różnorakie indeksy (zwane deskryptorami) dla danych multimedialnych. Najistotniejsze, z punktu widzenia problemu opisanego powyżej,

są deskryptory koloru i tekstury [2,5,6,8,9,10]. W zadaniach, które ukierunkowane są na poszukiwanie obiektów wydzielonych ze sceny istotne są natomiast deskryptory kształtu [1].

2. Opis problemu

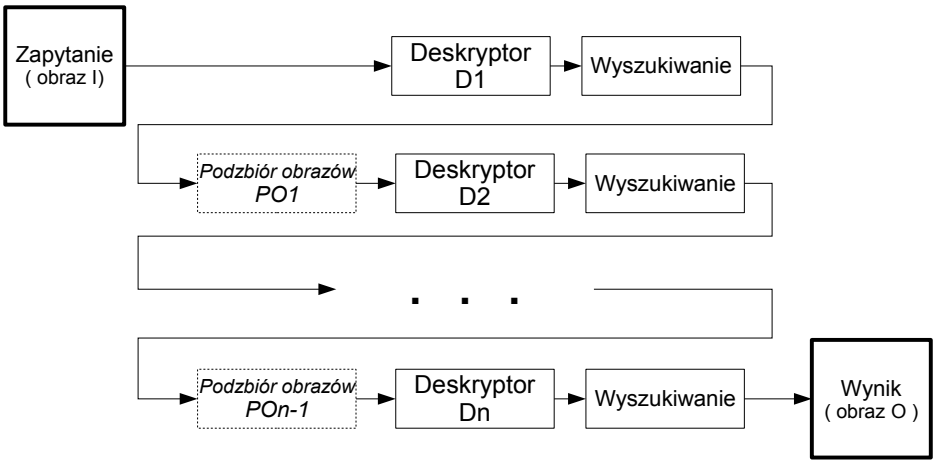
System indeksujący obrazy można opisać jako magazynujący informacje o zawartości obrazów a następnie przeszukujący stworzoną bazę danych (poprzez porównywanie indeksów obrazów). W literaturze systemy tego typu zaliczane są do klasy CBIR (ang. Content-Based Image Retrieval). Przykładowy schemat działania takiego systemu przedstawiony jest na rys. 1. Szybkość jego działania zależy głównie od wydajności bloku ekstrakcji cech i porównywania. O efektywności natomiast decyduje odpowiedni dobór cech obrazów.



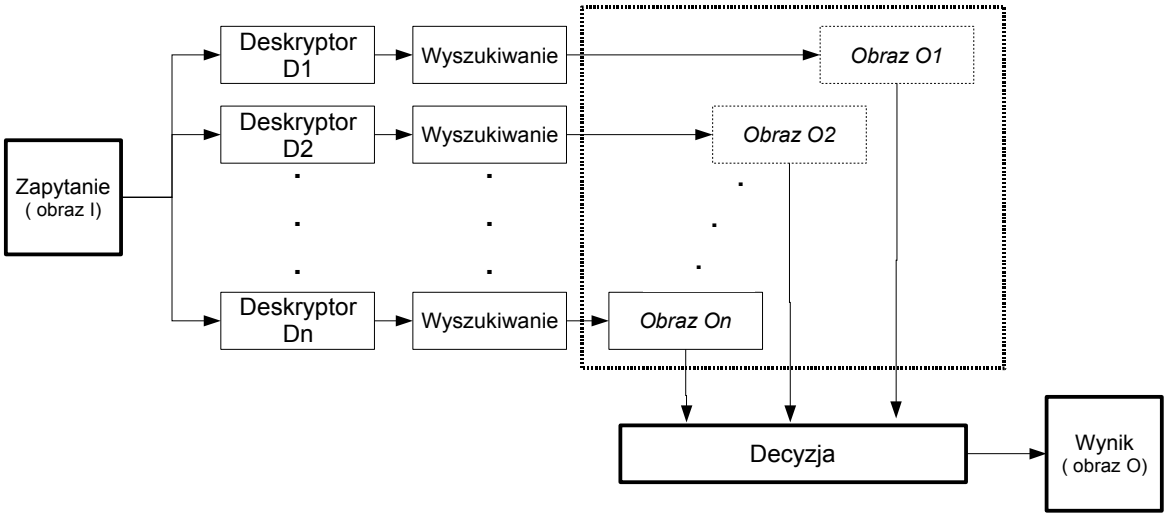
Rys. 1 *Schemat działania systemu CBIR*

Specyfika dużych zbiorów obrazów (liczących kilka tysięcy lub więcej obiektów) polega na tym, iż istnieje duże prawdopodobieństwo, iż użycie jednego tylko indeksu (deskryptora) nie zagwarantuje poprawnego wyniku wyszukiwania. Dlatego ważne jest łączenie deskryptorów w celu zwiększenia skuteczności tego procesu. Idea łączenia deskryptorów w powiązane struktury nie jest nowa. Jej zastosowanie jest szerokie i obejmuje między innymi złożone systemy identyfikacji biometrycznej [4].

Łączenie deskryptorów może odbywać się w sposób kaskadowy lub równoległy[4]. Pierwszy sposób (rys. 2) polega na tym, że wyniki wyszukiwania dla pierwszego deskryptora (D_1) i całego dostępnego zbioru obrazów są sortowane i ograniczane (z całego zbioru wydzielany jest podzbiór PO_1) a następnie przekazywane do kolejnego przeszukiwania względem drugiego deskryptora (D_2). Proces sortowania i ograniczania zbioru jest powtarzany tyle razy, ile deskryptorów liczy kaskada. Drugi sposób (rys. 3) zakłada równoczesne wyszukiwanie względem każdego z deskryptorów. Końcowy rezultat jest uzyskiwany za pomocą analizy wyników cząstkowych dla poszczególnych deskryptorów. Odbywać się to może metodą głosowania lub nadawania wag poszczególnym wynikom.



Rys. 2 *Kaskadowy sposób łączenia deskryptorów*



Rys. 3 *Równoległy sposób łączenia deskryptorów*

Każdy z powyższych schematów łączenia deskryptorów ma swoje zalety i wady. Niewątpliwą przewagą schematu kaskadowego jest jego intuicyjność. Z drugiej strony schemat równoległy pozwala na uniknięcie sytuacji, w której poprawnie wyszukany obraz (względem początkowego deskryptora) zostanie wyeliminowany w kolejnych etapach i nie zostanie uwzględniony w końcowym wyniku.

Przeprowadzona przez autorów analiza problemu spowodowała, że w opracowanym oprogramowaniu został wykorzystany kaskadowy schemat łączenia deskryptorów.

2.1 Deskryptory kolorowego obrazu statycznego

Do przeprowadzenia badań wybrane zostały następujące deskryptory należące do standardu MPEG-7 [8, 9, 10]:

- Scalable Color (SCD),
- Dominant Color (DCD),

- Color Layout (CLD),
- Edge Histogram (EHD),

oraz opracowane deskryptory bazujące na standardowych własnościach obrazu cyfrowego:

- histogram kolorów RGB (HIST),
- miniaturę obrazu o wymiarach 8 x 8 pikseli w odcieniach szarości (IBOX8),
- złożone trzy miniatury obrazu o wymiarach 8 x 8 pikseli dla przestrzeni RGB (RGBBOX8),
- wektor zawierający średnią wartość koloru RGB i dodatkowo średnią jasność (szarość) w obrazie (RGBI),
- wektor zawierający dominującą barwę (H) w modelu koloru HSV i dominującą jasność (V) (DHV).

Wszystkie deskryptory zostały zaimplementowane w prototypowych programach, które posłużyły do przeprowadzenia badań (nad wyszukiwaniem obrazów podobnych i tworzeniem foto-mozaik).

Deskryptor *Scalable Color*

Jest to jeden z deskryptorów zaproponowanych przez standard MPEG-7. Bazuje na przestrzeni barw HSV i transformacji Haara zastosowanej na wartościach histogramu koloru. Wartości histogramu są normalizowane a następnie wykonana jest na nich transformacja Haara. Proces tworzenia deskryptora składa się z następujących kroków [6, 8, 9]:

- zamiana przestrzeni kolorów obrazu RGB na model HSV,
- obliczenie znormalizowanego histogramu składowej H,
- transformata Haara na wektorze histogramu,
- zapis wartości transformaty do wektora cech.

Deskryptor *Dominant Color*

Wyznaczenie kolorów dominujących w obrazie dostarcza zwięzłego i prostego w implementacji wektora cech. Wykorzystany w badaniach uproszczony deskryptor oznaczony jako *F* opisany jest w sposób następujący:

$$F = \{\{c_i, p_i\}, \quad i = 1, \ldots, N\},$$

gdzie: N jest ilością kolorów, c_i jest trójelementowym wektorem koloru RGB, p_i reprezentuje udział koloru i w obrazie ($\sum_{i=1}^{N} p_i = 1$).

Wyznaczanie 8 kolorów dominujących dla obrazu odbywa się za pomocą algorytmu K-Średnich. Tworzenie deskryptora realizowane jest w następujących krokach [6, 9]:

- wyznaczenie kolorów dominujących (przypisanie wszystkich elementów obrazu do jednego z ośmiu podzbiorów),
- obliczenie procentowego udziału dla każdego koloru dominującego,
- zapis do wektora cech wszystkich kolorów dominujących oraz ich procentowego udziału.

Deskryptor *Color Layout*

CLD został zaprojektowany, by uchwycić rozkład przestrzenny kolorów w obrazie. Deskryptor sprawdza się bardzo dobrze w bazach danych zawierających szkice. W standardzie MPEG-7 sugerowane jest użycie 64-kolorowej reprezentacji obrazu w celu uproszczenia obliczeń. Deskryptor ten jest niewrażliwy na zmiany rozdzielczości i pozwala na bardzo szybkie wyszukiwanie obrazów. Wektor wynikowy składa się z próbek transformaty DCT dla poszczególnych składowych przestrzeni YC_bC_r. Budowanie deskryptora odbywa się w następujących krokach [6, 8, 9]:

- podział obrazu na 64 równe bloki niezależnie od jego rozdzielczości,
- znalezienie dla każdego bloku koloru reprezentatywnego i przedstawienie obrazu jako 64 elementy o reprezentatywnych kolorach,
- zamiana przestrzeni barw na YC_bC_r,
- obliczenie współczynników DCT niezależnie dla każdej składowej Y, C_b, C_r,
- uporządkowanie współczynników zgodnie z algorytmem "zig-zag" zdefiniowanym w standardzie JPEG,
- kwantyzacja współczynników DCT(Y) do 64 poziomów, DCT(C_b) i DCT(C_r) do 32 poziomów,
- zapis współczynników do wektora cech.

Deskryptor *Edge Histogram*

Deskryptor ten opisuje rozkład i kierunkowość krawędzi w podobszarach obrazu. Zakłada się, że rozróżniane są cztery rodzaje krawędzi kierunkowych: pionowa, pozioma, nachylona pod kątem 45 stopni, nachylona pod katem 135 stopni oraz krawędź bezkierunkowa. Tworzenie deskryptora przebiega następująco [6, 8, 9]:

- przekształcenie obrazu do odcieni szarości,
- podział obrazu na 16 bloków o równej wielkości,
- podział każdego z bloków na 4 subbloki,
- przefiltrowanie subbloków filtrami krawędziowymi w celu wykrycia krawędzi,
- obliczenie dla każdego z bloku mocy każdej z krawędzi,
- zapisanie częstości występowania krawędzi w obrazie w formie histogramu,
- zapis histogramu do wektora cech.

Znormalizowany histogram RGB (*HIST*)

Tradycyjny histogram kolorów w przestrzeni RGB jest bardzo często używany w systemach indeksowania obrazów. Jest prosty w implementacji i w bardzo zwięzły sposób reprezentuje zawartość obrazu. Tworzenie deskryptora odbywa się w następujących krokach:

- niezależne obliczenie histogramu dla każdego z kanałów RGB,
- podział każdego z histogramów na przedziały,
- obliczenie procentowego udziału pikseli w obrazie dla każdego z przedziałów w histogramach (normalizacja histogramu),
- zapisanie procentowego udziału pikseli dla każdego przedziału z każdego kanału.

Deskryptory typu *BOX8*

Deskryptory IBOX8 i RGBBOX8 są rozwiniętymi w wektory miniaturami obrazu o wymiarach 8 x 8 pikseli. Pomimo tego, iż są proste w implementacji, dają zaskakująco dobre

wyniki. W odróżnieniu od histogramu RGB, deskryptory te biorą pod uwagę informacje przez histogram nieuwzględnione, a mianowicie strukturę przestrzenną koloru w badanym obrazie. Deskryptory te są niewrażliwe na zmiany rozdzielczości obrazu i obecność niewielkiego zaszumienia. Wspólny schemat tworzenia deskryptora polega na:

- przeskalowaniu obrazu wejściowego do rozmiarów 8 x 8 (z uwzględnieniem interpolacji dwuliniowej),
- zapisanie 3x64 wartości (dla każdego z kanałów RGB) dla deskryptora RGBBOX8 i 64 wartości (jasności) dla IBOX8.

Deskryptor *RGBI*

Deskryptor ten jest w formie wektora zawierającego średnią wartość koloru RGB i dodatkowo średni poziom jasności w obrazie. Jest to prosty deskryptor pełniący rolę filtra wstępnie odrzucającego obrazy na początkowym etapie wyszukiwania.

Deskryptor *DHV*

Deskryptor DHV różni się znacząco od RGBI, ponieważ zawiera dominującą (a nie średnią) barwę (H) w modelu koloru HSV i dominującą jasność (V) (DHV). Pełni rolę podobną do RGBI.

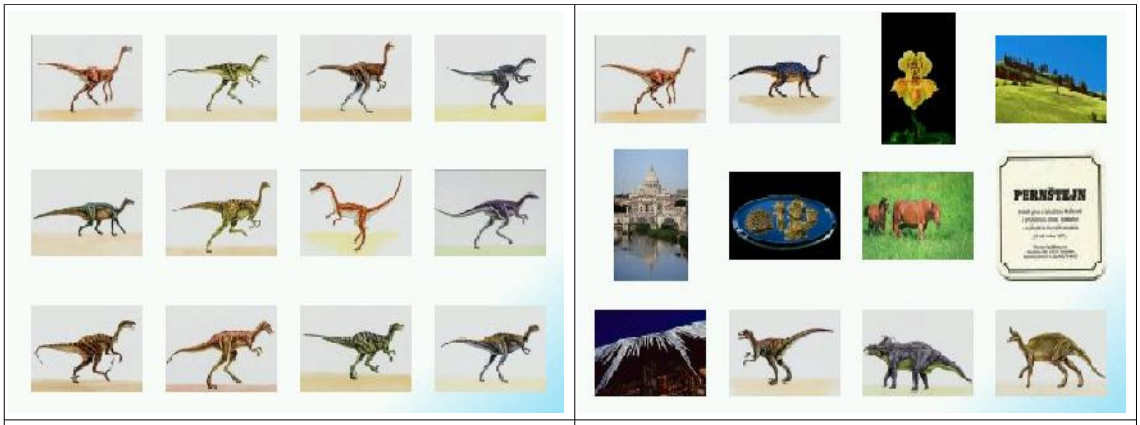
3. Oprogramowanie do indeksowania obrazów statycznych

W ramach badań stworzono oprogramowanie *Fast Image Finder* przeznaczone do wyszukiwania obrazów podobnych do wzorca z uwzględnieniem wybranych przez użytkownika deskryptorów (rys. 4). Deskryptory są wyliczane na podstawie informacji o kolorach i teksturze. Użytkownik może ingerować w parametry wyszukiwania poprzez ustawienie maksymalnej dopuszczalnej odległości pomiędzy deskryptorami.

Na wejściu do programu użytkownik zadaje przykładowy (wzorcowy) obraz. Wynikiem jest zbiór obrazów posortowanych od najbardziej do najmniej podobnego. Program działa na zasadzie analogicznej do znanych systemów CBIR takich jak VisualSeek i QBIC [7,11]. Zaimplementowano w nim kaskadowy sposób łączenia deskryptorów, który pozwala na intuicyjne tworzenie różnorakich schematów wyszukiwania. W programie dostępne są deskryptory: SCD, DCD, CLD, EHD, HIST i RGBBOX8.

Rys. 4 *Główne okno programu Fast Image Finder*

Opracowany program został przetestowany na zbiorze 5300 obrazów o różnorakiej tematyce i charakterze. Strategia badawcza polegała na wyszukiwaniu obrazów na podstawie wzorca. Ocenie podlegała ilość zwróconych obrazów zgodnych z oczekiwaniami użytkownika w stosunku do wszystkich obrazów wynikowych (metoda "ground truth"). Oprócz pierwszego zwróconego obrazu, pod uwagę brany był podzbiór pierwszych 2, 5 i 10 obrazów. Poniżej (rys. 5 i 6) przedstawiono przykładowe wyniki dla dwóch obrazów wzorcowych: "dino" (dominująca informacja o krawędziach) i "kwiat" (dominująca informacja o kolorze).



RGBBOX8	EHD

Rys. 5 *Wyniki wyszukiwania dla obrazu wzorcowego "dino"*

Poniżej (Tabela 1) zaprezentowano porównanie skuteczności wyszukiwania dla obrazu wzorcowego "dino". Zauważalna jest wysoka skuteczność prawie wszystkich badanych deskryptorów, za wyjątkiem EHD. Spowodowane jest to obecnością podobnych krawędzi we wszystkich zwróconych obrazach. Dodanie do niego, jako drugiego w kaskadzie, DCD, który uwzględnia informację o kolorze, poprawia znacząco skuteczność wyszukiwania.

Tab. 1 *Skuteczność wyszukiwania dla obrazu "dino"*

Deskryptor / / ilość obrazów branych pod uwagę	1	2	5	10
RGBBOX8	100%	100%	100%	100%
RGBBOX8+DCD	100%	100%	100%	100%
HIST	100%	100%	100%	100%
SCD	100%	100%	100%	100%
DCD	100%	100%	100%	100%
CLD	100%	100%	100%	100%
EHD	100%	100%	40%	30%
EHD+DCD	100%	100%	100%	100%

Wyniki dla wyszukiwania obrazów podobnych do wzorcowego obrazu "kwiat" (Tabela 2) potwierdzają wysoką skuteczność prawie wszystkich badanych deskryptorów. Z uwagi na fakt, iż w bazie znajdują się obrazy o podobnej kolorystyce, skuteczność dla deskryptorów

uwzględniających wyłącznie tę informację jest niższa. Najwyższą skutecznością wykazał się deskryptor RGBBOX8, gdyż jedynie on uwzględnia informację zarówno o kolorze, jak i jego rozkładzie przestrzennym. Najniższa skuteczność została zarejestrowana dla deskryptora HIST. Niewielka poprawa nastąpiła po połączeniu z nim EHD.

RGBBOX8 | HIST

Rys. 6 *Wyniki wyszukiwania dla obrazu wzorcowego "kwiat"*

Tab. 2 *Skuteczność wyszukiwania dla obrazu "kwiat"*

Deskryptor / / ilość obrazów branych pod uwagę	1	2	5	10
RGBBOX8	100%	100%	100%	100%
HIST	100%	100%	60%	30%
EHD+HIST	100%	100%	80%	40%
SCD	100%	100%	60%	70%
SCD+EHD	100%	100%	60%	70%
DCD	100%	100%	40%	50%
CLD	100%	100%	100%	80%
CLD+EHD	100%	100%	100%	90%

Nawet pobieżna analiza wyników pokazuje, że skuteczność wyszukiwania maleje wraz ze wzrostem ilości branych pod uwagę obrazów. Przyczyną takiego stanu rzeczy jest to, że w bazie znajduje się wiele obrazów o podobnych cechach niskopoziomowych (kolor i tekstura), natomiast żadna z metod nie odzwierciedla całkowicie percepcji człowieka. Każda z nich reaguje wyłącznie na wybrany zakres cech obrazu. Dlatego też najwyższą skutecznością charakteryzowały się deskryptory wiążące kolor i wzór (RGBBOX8) lub kaskady (CLD+EHD, EHD+ECD).

Nie zawsze jednak użycie deskryptorów typu RGBBOX8 jest możliwe. Często pojawia się problem wyszukiwania obrazu nie na podstawie przykładu, ale na podstawie opisu, charakterystyki, cechy typu kolor lub złożoność obrazu (tekstura). W takim przypadku jedynym dopuszczalnym scenariuszem jest użycie deskryptorów typu SCD, DCD i EHD. Poza tym deskryptory te są wyjątkowo "kompaktowe", w przeciwieństwie do miniatur obrazu.

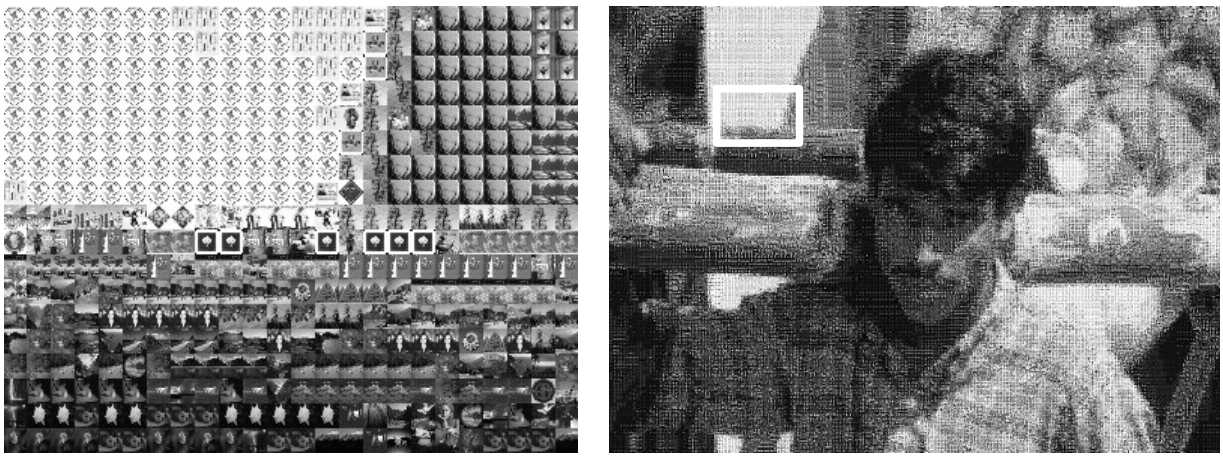
4. Oprogramowanie tworzące foto-mozaiki

Innym zastosowaniem indeksacji obrazów przy użyciu omawianych deskryptorów jest tworzenie foto-mozaik, czyli obrazów stworzonych z innych, mniejszych, obrazów. Ogólna zasada ich tworzenia mówi, że obraz bazowy o stosunkowo dużych rozmiarach składa się z relatywnie małych "kafelków" (ang. *tiles*). Kafelki pochodzą z dużego zbioru obrazów o możliwie szerokim spektrum zmian koloru i tekstury. Przy założeniu, że obraz bazowy ma wymiar 2048 x 1536 pikseli, rozmiar pojedynczego kafelka to 16 x 16 lub 32 x 32 piksele. Proces tworzenia mozaiki składa się z dwóch operacji powtarzanych cyklicznie dla każdego fragmentu obrazu:

- wyszukanie w bazie obrazów elementu o możliwie największym podobieństwie do wybranego fragmentu,
- zastąpienie w obrazie docelowym fragmentu wejściowego przez wybrany z bazy obraz.

W ramach badań stworzono w środowisku MATLAB prototypowe oprogramowanie do tworzenia mozaik. Z uwagi na odmienny od poprzedniego charakter zadania zaimplementowano w nim następujące deskryptory: HIST, IBOX8, RGBI i DHV. Przykładowy wynik tworzenia foto-mozaiki przedstawiony został na rys. 7.

Badania pokazały, że przy tworzeniu mozaik najistotniejsze są informacje dotyczące koloru obrazu. Mniej istotne są natomiast informacje o teksturze, gdyż stworzony obraz (mozaika) oglądany jest z reguły z dużej odległości i informacja o szczegółach jest w tej skali utracona. Z powodów wydajnościowych do końcowej implementacji wykorzystano deskryptory IBOX8 i RGBI ułożone w dwustopniową kaskadę.



***Rys.** 7 Wybrany powiększony fragment (z lewej) utworzonej mozaiki (z prawej)*

5. Podsumowanie

Zaprezentowane w pracy deskryptory w efektywny sposób opisują kolorowe obrazy statyczne. Ich charakter pozwala na szeroki zakres zastosowań, który nie jest ograniczony wyłącznie do standardowego indeksowania i wyszukiwania obrazów w dużych zbiorach. Zrealizowane badania potwierdziły, że opis uwzględniający informację o kolorze obrazu nie jest wystarczający do skutecznego rozwiązania problemu indeksowania i tworzenia mozaik. Dlatego zaproponowano uzupełnienie opisu treści obrazu za pomocą uproszczonej informacji o strukturze kolorów w obrazie (lub jego krawędziach), co zwiększyło znacząco użyteczność opracowanego oprogramowania.

Optymalnym, z punktu widzenia zadania indeksowania dużych zbiorów obrazów, jest użycie deskryptorów złożonych (RGBBOX8) lub kaskady deskryptorów (CLD+EHD). Odzwierciedla to w dużym stopniu proces wyszukiwania przez człowieka obrazów na podstawie przykładów. Natomiast w zadaniach tworzenia mozaik najważniejszy jest dobór fragmentu o podobnym charakterze, a nie treści, tak więc tutaj najlepiej sprawdzają się deskryptory uproszczone (IBOX8 i RGBI).

Literatura

1. Bober M., "MPEG-7 Visual Shape Descriptors", IEEE Transactions on Circuits and Systems for Video Technology, vol. 11, no. 6 , 2001
2. Deng Y., Manjunath B. S., Kenney C., Moore M. S., Shin H., "An Efficient Color Representation for Image Retrieval", IEEE Transactions on Image Processing,vol. 10, no. 1, 2001
3. Kukharev G., Forczmański P. "Hierarchical Method of Reduction of Features Dimensionality For Image Recognition And Graphical Data Retrieval", Pattern Recognition and Information Processing" Proceedings of Sixth International Conference PRIP'2001, Minsk, Republic of Belarus, 18-20 V 2001
4. Kukharev G., Kuźminski A., Nowosielski A., "Structure and Characteristics of Face Recognition Systems", "Computing, Multimedia and Intelligent Techniques", vol. 1 pp. 173-180, Institute of Computer and Information Sciences, Czestochowa University of Technology, 2005
5. Manjunath B. S., Ohm J.-R., Vasudevan V. V., Yamada A., "Color and Texture Descriptors", IEEE Transactions on Circuits and Systems for Video Technology, vol. 11, no. 6, 2001
6. Martínez J. M., "Overview of the MPEG-7 Standard. ISO/IEC JTC1/SC29/WG11 N4980", (Klagenfurt Meeting), http://mpeg.telecomitalialab.com/standards/mpeg-7/mpeg-7.html. 2002
7. Niblack W., Barber R., Equitz W., Flickner M., Glasman E., Petkovic D., Yanker P., Faloutsos P., Taubin G., "The qbic project: Quering images by content using color, texture, and shape", Poceedings of the SPIE Conference on Storage and Retrieval for Image and Video Databases, 2-3 II 1993, San Jose, CA, pp. 173-187, 1993
8. Sikora T., "The MPEG-7 Visual Standard for Content Description—An Overview", IEEE Transactions on Circuits and Systems for Video Technology, Vol. 11, No. 6, 2001
9. Skarbek W., "MPEG-7", IX Konferencja PLOUG, Kościelisko, X 2003
10. Snitkowska E., "Deskryptory tekstury w standardzie MPEG-7", IX Międzynarodowe Sympozjum Inżynierii i Reżyserii Dźwięku ISSET 2001, Warszawa, 18-20 X 2001
11. Smith J. R., Chang S.-F., "VisualSeek: a fully automated content-based image query system", Department of Electrical Engineering and Center for Image Technology for New Media,Columbia University, New York, 1996
12. Volmer S., "Tracing images in large databases by comparison of wavelet fingerprints", Proceedings of the 2nd International Conference on Visual Information Systems, San Diego, XII 1997, pp. 163-172, 1997
13. Wu Y.-G., Liu J.-H., "Image Indexing in DCT Domain", pp. 401-406, Third International Conference on Information Technology and Applications (ICITA'05) vol. 2, 2005