지역 특징 히스토그램 기반 영상식별자와 GPU 가속화

Image Identifier based on Local Feature's Histogram and Acceleration Technique using GPU

전 혁 준 † 서 용 석 †† 황 치 정 †††
(Hyeokjune Jeon) (Yongseok Seo) (Chijung Hwang)

요 약 현대의 대량화된 영상 관리 시스템은 영상의 특징을 표현하는 영상식별자에 대해 왜곡에 강인하며 빠른 검색 속도, 정확성 및 효율적인 저장 등의 기본 성능을 요구한다. 영상식별자 설계 방법은 기하학적 왜곡에 강인한 지역 방식과 빠른 검색 및 적은 저장 용량의 속성을 지닌 전역방식으로 구분 할 수 있다. 본 논문에서는 왜곡에 강하고 지역적 공간적 제약으로 인한 서로간의 차별성이 강화된 지역 기술자들로부터 각각 개개 차원의 특징 분포도를 분석하여, 두 영상간의 유사도를 빠르고 정확하게 측정할 수 있는 지역 기술자 및 전역 기술자의 속성을 가지고 있는 LFH(Local Feature's Histogram)기반 영상식별자를 제안한다. 또한 GPU를 사용하여 LFH를 구현하는 방법을 제시하며, 제안한 LFH와 대표적인 지역, 전역 방식인 SIFT 및 EHD 방식과 저장용량, 추출 시간, 검색 속도 및 정확률에 대한 성능을 비교하였다.

키워드 : LFH, 특징 히스토그램, 영상 식별자, 영상 디스크립터, 영상 검색

Abstract Recently, a cutting-edge large-scale image database system has demanded these attributes: search with alarming speed, performs with high accuracy, archives efficiently and much more. An image identifier (descriptor) is for measuring the similarity of two images which plays an important role in this system. The extraction method of an image identifier can be roughly classified into two methods: a local and global method. In this paper, the proposed image identifier, LFH(Local Feature's Histogram), is obtained by a histogram of robust and distinctive local descriptors (features) constrained by a district sub-division of a local region. Furthermore, LFH has not only the properties of a local and global descriptor, but also can perform calculations at a magnificent clip to determine distance with pinpoint accuracy. Additionally, we suggested a way to extract LFH via GPU (OpenGL and GLSL). In this experiment, we have compared the LFH with SIFT (local method) and EHD (global method) via storage capacity, extraction and retrieval time along with accuracy.

Key words : LFH, feature histogram, image identifier, image descriptor, image retrieval

1. 서 론

인터넷과 멀티미디어 기기들의 발전에 따른 고용량의 멀티미디어 데이터 전송과 대량화로 인하여 많은 분야에서 효율적인 관리(검색, 삭제, 색인, 중복처리, 저작권 처리 등)를 요구하게 되었다. 특히 개인 저작권 관련 영상들은 일부 내용이 왜곡(회전, 잘림, 이동, 압축, 밝기 변화 등)되어 저장, 유포되고 있어 대량화된 저작권 영상 관리에 있어 일부 손실, 왜곡이 발생한 영상들 간에 거리(유사도)를 빠르고, 정확하고 저용량으로 측정할 수 있는 영상 식별자는 매우 중요한 역할을 한다.

영상 식별자의 추출 방식은 크게 전역 방식과 지역 방식으로 구분할 수 있다. 전역 기술자(EHD(Edge

† 학생회원 : 충남대학교 컴퓨터공학부
fantajeon@cnu.ac.kr
†† 비 회 원 : 한국전자통신연구원 콘텐츠연구본부
yongseok@etri.re.kr
††† 종신회원 : 충남대학교 컴퓨터공학부 교수
cjhwang@cnu.ac.kr
논문접수 : 2010년 3월 10일
심사완료 : 2010년 7월 7일

정보과학회논문지: 컴퓨팅의 실제 및 레터 제16권 제9호(2010.9)

Histogram Descriptor[1]), Color Histogram, Trace Transform[2] 등) 추출 방식은 전체 영상을 대상으로 특징을 추출하거나 EHD 방식처럼 전체 영상을 다수개의 부-블록으로 나누고, 각각 부-블록에 속한 5가지 에지타입의(수평, 수직, 45도 대각선, 135도 대각선과 비방향성 에지) 분포도를 계산하여, 영상전체의 특징을 추출한다. 전역 방식의 영상 식별자는 빠른 거리 계산과 저용량의 장점을 가지고 있지만 사용자가 디지털적으로 왜곡할 수 있는 변환들인 회전, 이동, 투영변환 등에 취약하다는 단점이 있다. 지역 기술자 추출 방식은 (GLOH[3], PCA_SIFT[4], MSER[5], SIFT[6] 등) 영상으로부터 특징이 될 수 있는 특징 영역들을 기반으로 영상 정합 정도를 측정하여 두 영상의 유사성 정도를 판단한다. SIFT는 scale-space이론을 근거로 특징점(feature point)과 특징영역(feature region)을 추출한 후 그 영역의 그래디언트(gradient) 분포를 지역 기술자로 사용하는 방법으로 밝기 변화, 영상압축(JPEG compression), 블러(blur), 스케일 변환(scale change), 잡음, 시점(viewpoint) 변환 등에 강인한 속성을 지닌다[3]. 지역 식별자를 사용하여 두 영상간의 유사도 측정은 두 영상사이에서 추출된 특징 영역들의 매칭 정도, 즉, 두 지역 기술자들의 유사성으로 판단하여, 두 영역사이에 있는 점들에 대응점들을 찾은 후 RANSAC[7]을 사용하여 대응점의 잡음(outlier)을 제거하여 매칭 정도로 유사도를 측정한다. 즉 이러한 대응점 문제를 해결하여 유사도를 측정한다. 지역기술자 추출방식은 각종 왜곡에 강인하고 부분 정합이 가능한 장점이 있지만 대량화된 저작권 영상 관리에 있어서 전역기술자 방법에 비해 대응점을 찾아야하는 많은 처리시간과 지역 기술자들을 저장하기 위해 상대적으로 많은 용량이 요구되기에 대용량 처리 경우 효율성에 문제점을 가지고 있다. 지역기술자의 단점은 대응점 문제를 해결하기 위하여 다양한 방법들(pyramid matching[8], spatial matching[9])이 제안되었다. 지역 방식과 전역 방식을 혼합한 영상 기술자들[10-13]도 일부 존재하며 대표적인 방법인 bag-of-words[11,13]는 사전에 정의된 코드북(codebook), visual words 또는 단어 사전(dictionary)을 생성하여 이 코드북의 발생 빈도를 계산하여 유사성을 측정한다. 그러나 사전에 잘 정의된 코드북을 생성하지 않으면 성능에 저하를 가져오며, 또한 코드북 생성을 위하여 일반적으로 k-means와 같은 군집화(클러스터링, clustering)를 행하지만, 영상 경우 생성할 코드북이 고차원이기 때문에 클러스터링에 많은 어려움이 따른다(차원의 저주, the curse of dimensionality[14]).

본 논문에서 제안한 영상식별자(LFH)는 전체 영상의 특징영역으로부터 왜곡에 강인한 지역 기술자를 추출하여, 지역기술자들의 통계적 속성을 활용함으로써, 전역 기술자의 장점과 지역기술자의 장점인 빠른 검색, 적은 저장용량 및 왜곡에(회전, 이동, 잘림, 가려짐, 투영변환 등) 강인하여, 저작권 영상 관리 시스템의 효율성을 개선하였다.

본 논문의 구성은 2장에서는 제안한 방법인 LFH와 GPU기반 가속화 방법을 소개하고. 3장에서는 실험에 사용된 영상 왜곡 방법과 EDH, SIFT와 LFH에 대해 저장용량, 속도 및 정확도들에 대한 성능비교에 대해, 4장에서 결론 및 향후 연구에 대해 기술한다.

2. 제안 LFH

제안 LFH 생성 방법은 왜곡에 안정적인 특징 영역으로부터 추출한 강인한 지역 기술자들을 기반으로 변환(회전, 확대, 축소, 이동 등)이나 왜곡(잡음, 밝기 변화, 영상압축 등)에 강인한 속성을 가진 특징공간에서 히스토그램을 생성하는 것이다. 생성한 히스토그램은 왜곡에 강하고 지역적 공간적 제약으로 인한 서로간의 차별성이 강화된 지역 기술자들로부터 각각 개개 차원의 특징 분포도를 분석하여, 두 영상간의 유사도를 빠르고 정확하게 측정할 수 있기에 전역 기술자의 속성을 가지고 있다.

본 장에서는 영상으로부터 특징영역 추출, 특징영역으로부터 강인한 지역기술자 추출, 특징공간에서 LFH 생성, 두 LFH들 간에 유사도 측정 방법, LFH의 GPU 구현 과정에 대해 기술한다.

2.1 특징 영역 추출

주어진 영상에서 특징영역이란 각종 변화, 왜곡에서도 특징을 동일하게 추출할 수 있는 영역을 의미한다[10]. 특징영역을 추출하는 방법은 우선 왜곡에 강인한 특징점 주위에 적당한 크기의 윈도우를 사용하여 주변 밝기 영역들을 추출하는 방식[3,4,15,16]과 직접적으로 특징영역을 추출하는 방식[5]등이 있다.

본 논문에서는 순서적이며 강인한 특징영역을 추출하기위하여 Harris Corner Detection[15]을 사용하여 특징점들을 추출한 후, 특징점들의 주변밝기 정보들을 특별한 처리 없이 고정 크기(21×21)의 윈도우를 씌워서 특징영역을 추출한다. 윈도우 크기는 주변 밝기 정보를 중심으로 능동적으로 결정을 할 수도 있지만[6,17-19], 큰 영역보다 상대적으로 작은 영역이 기하학적 왜곡에 영향을 적게 받기에[20,21] 고정 크기이며 작은 크기의 특징영역을 추출하였다.

Harris Corner Detection은 주변 밝기 분포율의 변화율을 측정하여, 각 화소들의 강도를 계산할 수 있으므로. 이 강도들로 특징점들의 추출 순서를 결정하며 영상 식별자를 강하게 만들기 위하여 인접한 특징점들을 삭제하는 방식(중복점 제거)으로 특징점들을 영상 전체에 가급적 고르게 분산시켰다.

2.2 지역기술자

특징영역에 대해 밝기 불변 처리, 회전 방향 할당(orient assignment)[6] 및 밝기 평균 부-블록 직렬화 과정을 거쳐 지역기술자를 추출한다.

밝기 불변 처리 단계는, 지역기술자가 밝기에 불변인 속성을 가지도록 전체영상으로부터 추출한 각각의 특징영역 $I(v)$에 식 (1)을 적용, $S(v)$로 변환시킨다.

$$S(v)=\frac{I(v)-\beta}{\alpha-\beta}\times C \begin{cases} \text{if } S(v)<0,\ S(v)=0 \\ \text{if } S(v)>C,\ S(v)=C \\ otherwise \quad S(v) \end{cases} \qquad (1)$$

여기서 $I(v)$를 밝기 순으로 오름차순 정렬을 하였을 때, α는 상위 k번째의 밝기 값(최소값)이고, β는 하위 k번째 밝기 값(최대값)이며. C는 사용자 상수 값으로 함수 $S(v)$가 가질 수 있는 최대 범위이다.

회전 방향 할당 단계는 회전 불변을 위하여 SIFT[6]의 방법을 사용하였다. SIFT에서 제안한 기본 방법인 방향 히스토그램(orient magnitude histogram)에서 최대값을 가지는 고유한 지역각(local orientation)을 찾아서, 수평축에 정렬하여 특징영역을 회전 불변으로 만든다.

밝기 평균 부-블록 직렬화 단계는 지역기술자의 차별성을 강화하기 위하여, 특징 영역을 부-블록화 한다. 이 방법은 각각 지역 기술자들 간에 차이성을 강화하기 위하여, 다른 기술자들(GOLH[3], SIFT[6] 등)이 주로 사용하는 방법으로 특징영역을 정규화 크기로 세부 분할한다(부-블록화). 즉, 다른 블록들 간의 발생 빈도에 지역적 기하학적인 제약 속성을 부여하여, 지역 기술자들 간에 차별성을 더 강화시킨다. 본 논문에서는 회전 불변된 특징영역을 5×5개의 부-블록들로 분할하고, 잡음에 강인한 각각 블록의 대표 값을 계산하기 위하여, 각 부-블록에 속해있는 화소들의 밝기 값들의 평균값을 대표 값으로 사용한다. 이후 각각의 부-블록들을 행(열) 우선순위 직렬화(순서화, serialization)를 거쳐서 25차원의 잡음, 밝기 및 회전에 강인한 속성을 가진 지역기술자를 추출 한다(그림 1 참조).

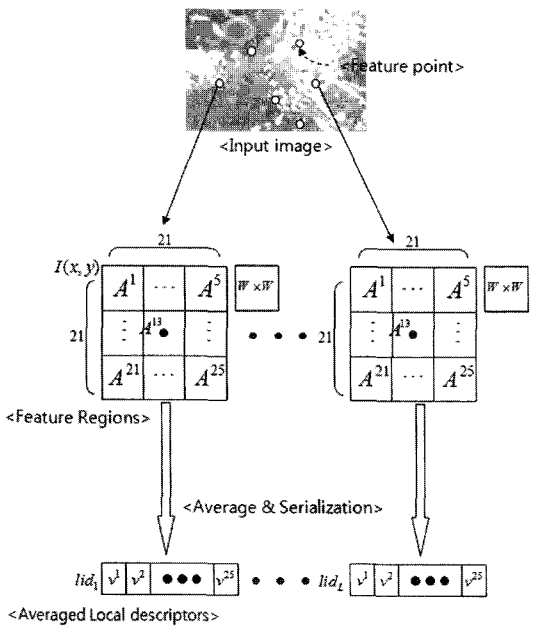


그림 1 25차원(5×5 부-블록) 지역기술자 추출과정

2.3 LFH 생성

영상의 특징영역으로부터 공격에 강인한 지역기술자들을 추출하면, 이 지역기술자들로부터 생성되는 새로운 고차원 공간인 특징공간(feature space)을 클러스터링과 같은 특별한 과정없이 각각의 차원에 대해 독립적으로 지역기술자들의 분포도(16빈 크기의 양자화된 히스토그램)를 생성한다. 영상으로부터 추출한 L개의 지역기술자들로 부터 총 25개의 히스토그램으로 구성된 LFH를 생성한다(그림 2 참조). 그림 2에서 지역기술자가 25차원이므로 각각의 히스토그램 $P(x)$의 크기가 16이므로 LFH는 1×16 크기인 총 25개의 히스토그램을 일렬로 결합한(나열한) 것과 같다. 즉 LFH는 $[P(x_1),P(x_2),\cdots,P(x_{25})]$인 400 차원의 벡터가 된다.

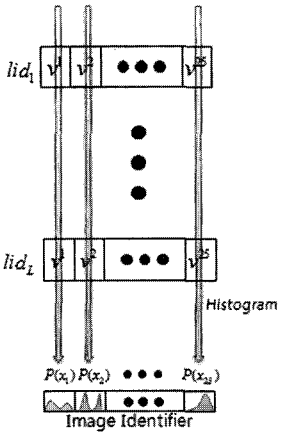
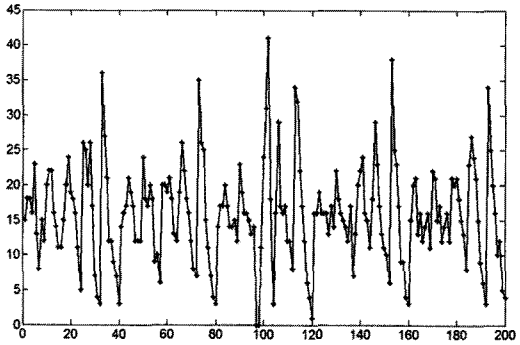


그림 2 25차원 지역기술자로부터 LFH 생성 과정

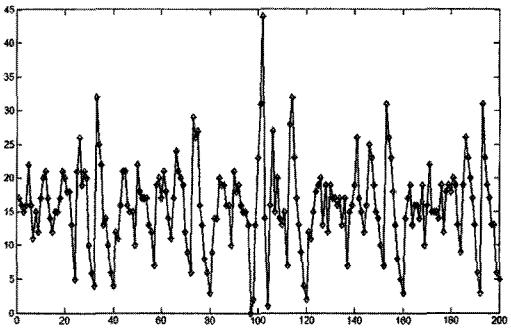
고차원의 LFH는 저장용량과 거리 계산 속도에 영향을 주므로 정확도는 크게 떨어지지 않는 범위에서 LFH의 저장 용량과 거리 계산 속도의 성능을 높이기 위하여 1차원 웨이블릿(wavelet)를 적용하여 원본 신호를 유지하며 잡음에 둔감한 영역인 저주파 영역으로 차원 축소된 200차원의 LFH를 생성한다. 차원 당 1바이트를 할당하여 영상 당 200바이트 크기의 LFH를 생성한다. 그림 3은 원 영상과 45도 회전된 왜곡영상에 대해 생성한 LFH를 보여주고 있다.

(a) 원본 영상

(b) 45도 시계방향 회전 영상



(c) (a)의 haar wavelet으로 차원 축소된 LFH



(d) (b)의 haar wavelet으로 차원 축소된 LFH

그림 3 원본 영상과 45도 회전영상의 LFH

2.4 거리(유사도) 계산

거리 계산 방식은 연산 속도와 정확도를 고려하여 결정을 한다. 일반적으로 히스토그램 방식에서는 유클리드 거리(식 (2)), 교차 면적 방법(식 (3))과 Chi-squared 거리(식 (4)) 방식을 따른다. 본 논문에서는 히스토그램에서 일반적으로 많이 사용하는 교차 면적 방법을 사용하여 두 LFH 간의 거리를 측정 및 유사도를 계산한다.

$$dist(A,B)=\sum_{i=1}^{N}(a_i-b_i)^2 \tag{2}$$

$$intersect(A,B)=\sum_{i=1}^{N}\min(a_i,b_i) \tag{3}$$

$$X^2(A,B)=\sum_{i=1}^{N}\frac{(a_i-b_i)^2}{2(a_i+b_i)} \tag{4}$$

2.5 GPU 가속화 구현

GPU 가속화는 GLSL(OpenGL Shading Language) [22]를 사용하여 영상 필터 처리를 구현하였다(그림 4 참조). OpenGL은 기본적으로 정점단위 처리인 정점 쉐

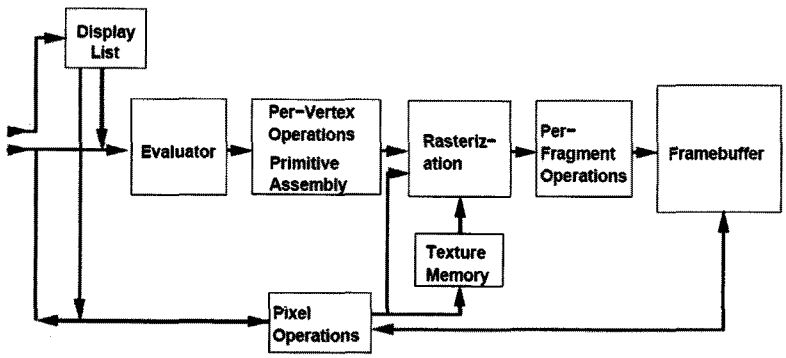


그림 4 OpenGL Rendering Pipeline[23]

이더(vertex shader, 그림 4의 per-vertex operations primitive assembly 블록에 존재)와 프래그먼트(실제 화소를 생성하기위한 정보를 담고 있는 추상적 화소) 처리 단위인 프래그먼트 쉐이더(fragment shader, 그림 4의 rasterization 블록에 존재)로 구성된다. 이 두 개의 쉐이더 코드는 GLSL을 사용하여 직접 작성할 수 있는 구조로서 모든 처리는 독립적인 병렬처리로 수행된다. 즉, 정점 쉐이더는 한 정점만 처리하도록 코드를 작성하고, 프래그먼트 쉐이더는 한 프래그먼트에 대해서만 독립적으로 처리하도록 코드를 작성한다.

OpenGL[23]의 OpenGL Rendering Pipeline의 정점 처리 과정(Per-vertex Operation Primitive Assembly)은 그림 5와 같다. 사용자가 OpenGL API를 사용하여 object coordinate인 정점, 4×4의 model-view matrix M, 4×4의 projection matrix P, viewport에서 변수 x, y, w, h, f와 n을 설정하면 최종적으로 window coordinates를 얻는다. 위 과정으로 계산된 좌표들은 이후 프래그먼트 쉐이더가 포함된 rasterization 과정에서 프래그먼트(framgment)가 생성되고, 최종 framebuffer(graphic card)에 2차원 화소 처리된 영상을 저장하게 된다.

영상 처리 목적으로 수행하기 위하여 정점 쉐이더는 OpenGL의 가지고 있는 기본 기능을 사용하고, GLSL을 사용하여 프래그먼트 쉐이더만을 프로그램 한다. 기본적인 정점 처리 과정은 model view matrix, projection matrix, perspective division, viewport transformation 과정을 수행한다(그림 5 참조). 정점 쉐이더의 기능은 영상처리 필터에서 사용하지 않으므로, object coordinates는 $(0,H_{img},0),(W_{img},H_{img},0),(W_{img},0,0),(0,0,0)$인 사각형 정점 정보, model view matrix M와 projection matrix P값은 모두 4×4 단위행렬로 처리하고, viewport transformation에서 x=0, y=0, w= W_{img}, h=H_{img}, f=1 그리고 n=0으로 설정한다. 여기서 H_{img}는 입력 영상의 높이, W_{img}는 입력 영상의 넓이이다. 화소단위 필터 처리를 위하여 프래그먼트 쉐이더만을 프로그램하여 영상 필터 처리를 수행한다. GPU에서 필터 연산을 수행할 때 CPU의 메인 메모리에 있는 영상을 GPU의 프래그먼트 쉐이더로 영상을 전송하는 방법은 OpenGL API를 이용하여 텍스쳐 메모리(texture memory)에 상주하는 텍스쳐(texture)로 영상을 전송한다(write 과정). 그리고 최종 필터 처리 결과는 RenderToTexture 기법을 사용하여 다른 텍스쳐에 저장하고 OpenGL API을 사용하여 다시 CPU의 메인 메모리로 적재한다(read 과정).

제안한 LFH를 GPU기반 가속화 과정은 그림 6과 같다. 그림 6에서는 CPU와 GPU 파트들로 구분을 한다. CPU에서 담당하는 과정은 제일 처음 저장 장치의 영상

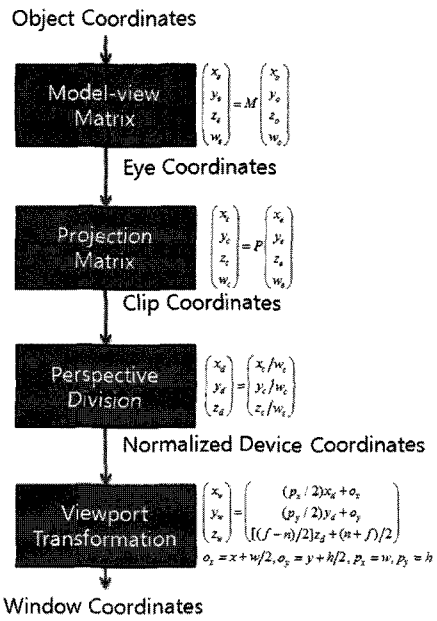


그림 5 정점 처리 과정[3]

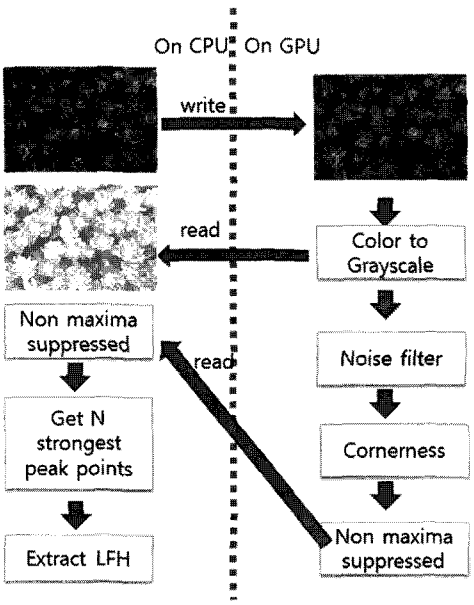


그림 6 CPU와 GPU 역할

을 주 메모리를 읽어들이는 과정, 직렬성 알고리즘과 GLSL에 기능 구현상 한계와 관련된 기능들을 처리하였다. 그 외에 GPU상에서 화소 단위 병렬로 구현 가능한 처리인 단색변환(Color to Grayscale), 잡음 제거(noise filter), Harris Corner Detection의 코너 강도 계산(Cornerness) 그리고 코너의 지역 최대값 추출(non-maxima

suppressed)을 GPU에서 구현하였다. 특히 지역 최대값 추출 과정 이후 특징점 순서 정렬, 히스토그램 및 기타 데이터 생성 같은 것들은 현재의 GPU와 GLSL에서는 구현하기에 어려운 구조이므로 최대값 추출 결과와 단색 영상을 CPU로 복사(read)하여 최종적으로 제안한 LFH를 추출하였다.

3. 실 험

본 장에서는 제안한 방법(LFH)과 EHD 및 SIFT의 속도, 저장 크기 및 정확도에 대해 Intel Pentium 4 Core 2 DUO E8400 @ 3.00GHz, nVidia Geforce 9800 GT Graphic card를 사용하여 비교 실험하였다.

3.1 왜곡영상

실험에 사용한 왜곡 영상들은 디지털 상에서 일반 사용자가 영상을 많이 왜곡시키는 12종류의 유형들에 대해 2-3개 정도로 왜곡 강도를 달리하여 사용하였다. 모든 원본 및 왜곡 영상의 기본 크기는 426×284이다. 그러나 비례 축소(size reduction) 및 장축 늘림(extension)은 각각 다르다. 비례 축소에서 10%는 383×255, 30%는 297×198이고 50%는 213×142이다. 마지막으로 수평 축 늘림은 1.33과 1.5는 각각 566×284와 639×284이다. 밝기 변화(bright)는 원 화소 값에 10%, 20%와 25%을 곱하여 왜곡시켰다. Grayscale은 컬러 영상을 그레이 영상으로 변환하였으며, jpeg는 80, 60 및 30의 압축률로 하였다. Gaussian은 표준편차의 값을 4, 8 및 12로 하여 잡음을 추가하였으며. Autolevel은 화소 값에서 하위 및 상위 0.1%에 해당되는 값들을 0과 255로 매핑 및 선형변환을 하였다. Blur는 마스크의 사이즈를 3, 5 및 7로 하였으며, 영상 회전(rotation)의 각은 90, 180, 270, 10, 25 및 45로 하였다. 기울어짐(shearing)은 수평축을 기준으로 수직축의 기울어지는 각으로 정의하며, 각각 4도, 6도 그리고 10도로 하였다. 투영변환(pers, perspective projection transform)은 원본 영상의 중심을 기준으로 영상의 상단은 앞쪽으로 하단은 뒤쪽으로 기울여지는 각이다. 즉, 영상의 평면을 X와 Y축으로 정의하였을 때, 깊이 축인 Z축의 반시계방향 회전각으로 각각 4도, 6도 그리고 10도로 왜곡하였다. 이동(trans, translation)은 원본 영상의 크기의 수평축과 수직축을 각각 10%, 20% 및 30% 비율로 원본 영상을 대각선 이동하였다.

본 실험의 기하학 공격 같은 투영변환, 기울어짐, 이동, 회전들의 제로패딩(zero-padding) 현상을 방지하기 위하여, 기본적으로 큰 영상으로부터 오려낼 영역을 정한다. 원본 영상은 손상 없이 큰 영상의 중심으로부터 수평축이 장축이면 426×284로 수직축이 장축이면 284×426으로 오려냈으며, 왜곡 영상들은 원본 큰 영상을 왜곡시킨 후 중심으로부터 같은 크기로 오려냈다(그림 7 참조). 그림 8은 수직축이 장축인 영상 경우 원본과 왜곡 영상들의 예를 보여준다.

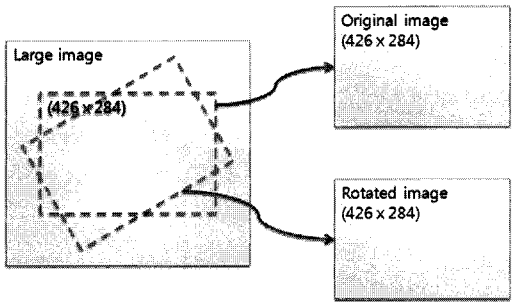


그림 7 원본 영상과 질의 영상 추출 과정

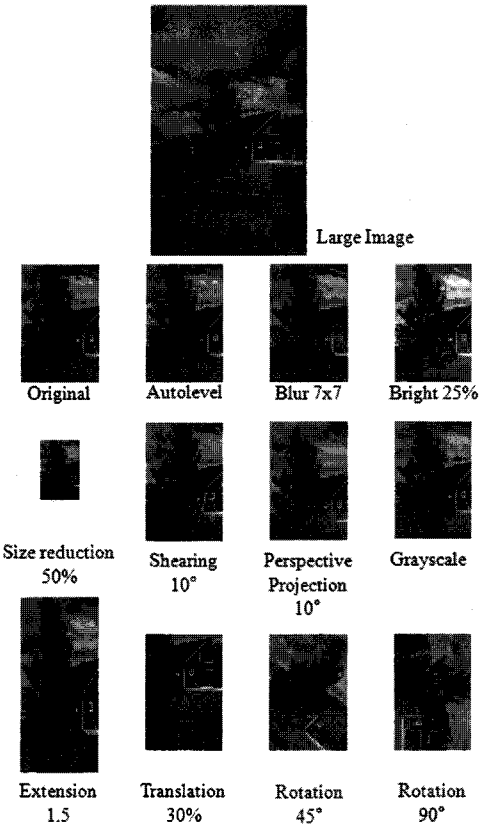


그림 8 원본 영상과 왜곡 영상 예시

3.2 속도비교

영상식별자 추출 및 거리 계산 속도 비교는 표 1과 같다. 표 1에서와 같이 추출 시간 및 거리 계산 속도는 EHD, LFH 그리고 SIFT 순이었으며, EHD와 LFH는 SSE2로 계산하였을 경우 EHD는 약 2.96배, LFH는 약 4.46배 정도 더 빨랐다. 여기에 비하면 SIFT의 추출 및

표 1 추출 시간 및 거리 계산속도

속도	EHD	SIFT	LFH
추출 시간 (sec)	0.004	2	1.62
거리 계산 (pairs/sec)	21,568,703 / 64,007,045(SSE2)	7	3,801,160 / 16,977,952(SSE2)

거리 계산 속도는 현저히 느렸다. 또한 LFH는 GPU로 가속되었을 경우 영상식별자 추출 시간이 0.079초로 GPU 구현이 CPU 구현보다 약 20배정도 빨랐다. GPU는 병렬처리 구조라서 실제 보다는 더 빠른 처리 시간을 가져야하지만 약 20배로 빠른 속도로 측정된 것은 총 처리 시간 중에서 CPU와 GPU간의 메모리 입출력 처리가 평균적으로 약 82.96% 이상 소비하였기 때문이다.

3.3 용량 비교

각각 영상식별자가 차지하는 용량을 계산하면, 표 2와 같다. EHD는 실수 형을 소수 점 6자리까지 절단하여 정수형(2^{32})으로 저장 및 실험하였으므로, 80차원 × 정수형(4바이트)이므로 총 320 바이트를 차지하였다. 그러나 EHD의 정확률은 떨어지나 각 차원당 1바이트로 할당하는 경우에는 80 바이트가 된다. LFH는 25(부-블록 개수)×16(빈 개수)×1바이트에서 wavelet 변환하여 총 200 바이트를 할당하였다. SIFT는 영상에서 추출한 특징점(keypoint)의 개수에 따라 저장용량이 가변적이며 평균 768,344 바이트가 할당 되였으며, 일부 JPEG 영상들은 저장 용량보다 약 10배는 더 많이 저장되었다. 그림 9는 SIFT 저장용량 분포도이다. 저장 용량의 최소 크기는 0 바이트(특징이 전혀 없는 영상)부터 최대 크기는 3,502,224 바이트까지 분포하였다.

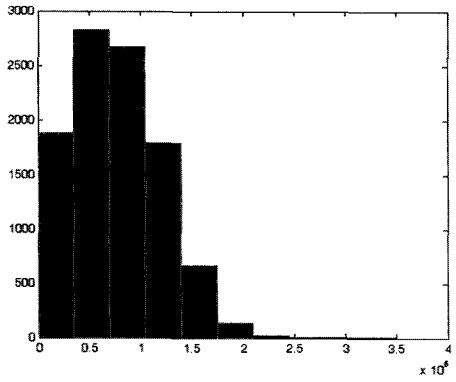


그림 9 SIFT용량 분포도

표 2 식별자 저장 용량 비교

	EHD	LFH	SIFT
용량(bytes)	80	200	768,344

표 3 정확도 비교표

공격종류	EHD	SIFT	LFH
bright10	100.00	97.50	99.50
bright20	99.50	96.00	98.25
bright25	99.00	94.00	95.50
grayscale	100.00	99.00	100.00
jpeg80	100.00	98.50	100.00
jpeg60	100.00	98.75	100.00
jpeg30	99.50	99.00	99.00
gaussian 4	99.50	98.50	99.50
gaussian 8	98.00	98.50	99.25
gaussian 12	95.50	98.00	98.00
autolevel	100.00	96.75	99.50
blur3x3	100.00	98.75	100.00
blur5x5	100.00	98.00	99.75
blur7x7	99.25	96.00	98.75
rotation 90	2.50	98.75	84.25
rotation 180	6.75	99.25	100.00
rotation 270	2.00	99.00	83.50
rotation 10	95.50	99.00	100.00
rotation 25	21.25	98.75	98.75
rotation 45	3.50	98.75	93.00
reduction 10	98.50	99.25	100.00
reduction 30	99.00	98.75	100.00
reduction 50	98.25	97.75	100.00
extension 1.33	97.25	99.25	100.00
extension 1.5	96.50	99.00	100.00
shearing 4	100.00	99.25	100.00
shearing 6	99.75	99.25	100.00
shearing 10	98.75	99.25	100.00
pers 4	97.75	99.00	99.75
pers 6	96.75	99.00	99.00
pers 10	92.00	98.50	95.75
trans 10	85.00	99.25	95.50
trans 20	20.25	99.00	77.25
trans 30	8.50	98.00	52.25
평 균	79.70	98.38	96.05

3.4 정확도

정확도 계산은 각각의 왜곡 당 원본과 질의 영상 400장으로 실험을 하였다. 정확률은 하나의 질의를 하였을 때 가장 유사하다고 판단한 것이 실제 원본과 같은 확률로 정의를 하였다(식 (5) 참조). 여기서 Q는 총 질의한 개수 이고, T는 1순위로 정확히 찾은 개수이다.

$$Accuracy(\%) = \frac{T}{Q} \times 100 \quad (5)$$

표 3에서와 같이 정확률은 전반적으로 SIFT가 가장 높았다. EHD와 LFH는 전역 속성을 가져서 SIFT보다는 정확률이 낮았지만, LFH는 전역 기술자 EHD보다는 회전과 이동 변환에서는 높은 성능을 보였다. LFH는 기하학 공격에 EHD보다는 강한고 다른 단순 왜곡들에

서는 비슷한 성능을 지녔다. 지역 기술자인 SIFT와 LFH를 비교하면 거의 모든 면에서 역시 비슷한 성능을 지녔으나, LFH가 회전과 이동면에서 일부 차이가 난다. 즉, LFH는 SIFT와 비교하여 많은 부분 비슷한 성능을 가졌으며, EHD와 비교하였을 때 거의 모든 면에서 동일한 정확률을 가졌지만, 기하학적 왜곡에서는 더 높은 정확률을 가졌다.

4. 결론 및 향후 연구

제안 LFH는 특징 영역으로부터 추출한 강인한 지역 기술자들을 기반으로 변환(회전, 확대, 축소, 이동 등)이나 왜곡(잡음, 밝기 변화, 영상압축 등)에 강인한 속성을 지닌 특징 공간에서의 히스토그램을 생성하는 것이다. 생성한 히스토그램은 왜곡에 강하고 지역적 공간적 제약으로 인한 서로간의 차별성이 강화된 지역 기술자들로부터 각각 개개 차원의 특징 분포도를 분석하여, 두 영상간의 유사도를 빠르고 정확하게 측정할 수 있기에 전역 기술자의 속성을 가지고 있다.

대량화된 저작권 영상 관리 시스템 구축 시 요구되는 처리 속도, 정확률과 영상 당 저장 용량 면에서 전역기술자방식인 EHD와 지역기술자방식인 SIFT와의 성능비교 실험 결과 LFH는 EHD에 비해 처리 속도 및 저장 용량은 크게 차이가 나지 않으며 특히 이동 변환(기하학적 왜곡)에 대해 높은 정확도를 가지고 있으며 SIFT와는 비슷한 정확도를 가졌으며 처리 속도 및 저장 용량에 대해선 매우 높은 성능을 보였다. 또한 LFH는 지역기술자들의 순서와 상관없이 생성되므로, SIFT와 같은 지역 기술자 기법들 경우 두 영상 간의 매칭을 위한 대응점 문제(지역 기술자들 간에 거리 비교와 대응점 내에 잡음(outlier) 제거를 위한 RANSAC 과정)를 회피할 수 있는 장점을 가지고 있어 LFH를 활용하여 효율성 높은 대량화된 영상 관리시스템구축이 가능하다.

향후 연구로는 제안한 LFH의 특징 공간을 생성하는 효과적인 지역 기술자를 개발하여 성능을 향상시키며 또한 LFH의 차원의 크기는 검색 속도와 연관이 되므로, 더 효율적인 차원 축소 방법에 대한 연구가 지속적으로 수행하여야 한다.

참 고 문 헌

[1] MPEG-7, "Text of ISO/IEC 15938-3/FDIS Information Technology - Multimedia Content Description Interface - Part 3 Visual," ISO/IEC JTC/SC29/WG11/N4358, Sydney, 2001.

[2] A. Kadyrov and M. Petrou, "The Trace Transform and Its Applications," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol.23, pp.811-828, 2001.

[3] K. Mikolajczyk and C. Schmid, "A Performance Evaluation of Local Descriptor," *IEEE Transaction on Pattern Analysis and Machine Intelligence*, vol.27, no.10, October 2005.

[4] Y. Ke and R. Sukthankar, "PCA-SIFT: A More Distinctive Representation for Local Image Descriptors," *Proc. Conf. Computer Vision and Pattern Recognition*, pp.511-517, 2004.

[5] J. Matas, O. Chum, M. Urban, and T. Pajdla, "Robust Wide Baseline Sereo from Maximally Stable Extremal Regions," *Proc. of British Machine Vision Conference*, pp.384-393, 2002.

[6] D.G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *International Journal of Computer Vision*, vol.20, pp.91-100, 2003.

[7] M.A. Fischler and R.C. Bolles, "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography," *Commun. ACM*, vol.24, no.6, pp.381-395, 2004.

[8] K. Grauman and T. Darrell, "The Pyramid Match Kernel: Discriminative Classification with Sets of Image Features," In *Proc. ICCV*, 2005.

[9] S. Lazebnik, C. Schmid and J. Ponce. "Beyond Bags of Features, Spatial Pyramid Matching for Recognizing Natural Scene Categories," *Proceedings of CVPR*, pp.2169-2178, 2006.

[10] H.J Jeon, J.K. Jeong, J.W. Bang, and C.J. Hwang, "Sparse Intensity Histogram: Distinctive and Robust to the Space-distortion," *ICAPR 2009*, pp. 53-56, 2009.

[11] L. Fei-Fei and P. Perona, "A Bayesian Hierarchical Model for Learning Natural Scene Categories," *Proc. of IEEE Computer Vision and Pattern Recognition*, pp.524-531, 2005.

[12] T. Leung and J. Malik, "Representing and Recognizing the Visual Appearance of Materials Using Three-dimensional Textons," *International Journal of Computer Vision*, vol.43, no.1, pp.29-44, 2001.

[13] T. Hofmann, "Unsupervised Learning by Probabilistic Latent Semantic Analysis," *Machine Learning*, vol.43, pp.177-196, 2001.

[14] R. Bellman, "Adaptive Control Processes: A Guided Tour," Princeton University Press, 1961.

[15] C. Harris and M. Stephens, "A Combined Corner and Edge Detector," *Fourth Alvey Vision Conference*, pp.147-151, 1988.

[16] D. Reisfeld, H. Wolfson and Y. Yeshurun, "Context-Free Attentional Operators: The Generalized Symmetry Transform," *International Journal of Computer Vision*, vol.14, pp.119-130, 1995.

[17] T. Lindeberg, "Feature Detection with Automatic Scale Selection," *International Journal of Computer Vision*, vol.30, pp.79-116, 1998.

[18] Y. Dufournaud, C. Schmid and R. Horaud, "Matching Images with Different Resolutions," *Computer*

Vision and Pattern Recognition, vol.1, pp.612-618, 2000.

[19] K. Mikolajczyk and C. Schmid "Indexing Based on Scale Invariant Interest Points," In *Proc. ICCV*, pp.525-531, 2001.

[20] B.D. Lucas and T. Kanade, "An Iterative Image Registration Technique with an Application to Stereo Vision," *IJCAI81*, pp.121-130, 1981.

[21] H.Y. Jeon, J.K. Jeong, J.W. Bang, and C.J. Hwang, "The Efficient Features for Tracking," *Twentieth International Conference on Tools with Artificial Intelligence*, vol.1, pp.241-244, November 2008.

[22] J. Kessenich, "The OpenGL Shading Language," http://www.opengl.org/registry/doc/GLSLangSpec.1.50.09.pdf, 2009.

[23] M. Segal and K. Akeley, "The OpenGL Graphics System: A Specification," http://www.opengl.org/registry/doc/glspec21.20061201.pdf, 2006.

전 혁 준

2004년 우송대학교 컴퓨터과학 학사. 2008년 충남대학교 컴퓨터공학 석사. 2008년~현재 충남대학교 컴퓨터 공학 박사과정. 관심분야는 패턴인식, 컴퓨터비전, 인공지능

서 용 석

1999년 영남대학교 전자공학과 공학사 2001년 영남대학교 정보통신공학과 공학석사. 2009년 충남대학교 컴퓨터공학과 공학박사. 2001년~현재 한국전자통신연구원 콘텐츠연구본부 선임연구원. 관심분야는 저작권 정보 보호, 신호처리 및 압축

황 치 정

1975년 서강대학교 수학과 학사. 1979년 서강대학교 수학과 석사. 1981년 뉴욕주립대학 수학과 석사. 1985년 University of Connecticut 전산학 석사. 1987년 University of Connecticut 전산학 박사 2006년~2010년: 충남대학교 차세대정보기술 SW인력양성사업단(BK21) 단장. 1988년~현재 충남대학교 컴퓨터공학과 교수. 관심분야는 영상처리, 패턴인식