논문 2009-46SP-5-8

스테레오 영상에서 움직임 벡터를 이용한 고속 변이 벡터 추정

(Fast Disparity Vector Estimation using Motion vector in Stereo Image Coding)

도 남 금*, 김 태 용**

(Nam-Keum Doh and Tae-Yong Kim)

요 약

스테레오 영상은 단일 영상과는 달리 오른쪽과 왼쪽, 2개의 영상으로 구성되어 있기 때문에 단일 영상에 비하여 더욱 많은 데이터량을 가지게 된다. 따라서 이를 효율적으로 처리하기 위한 영상 압축 기술이 필요하게 되었고, 이를 위해 DPCM기반의 예측 부호화 압축 기술을 대부분의 비디오 압축 표준에서 사용한다. 예측 부호화 기술의 구현을 위해 움직임 추정 및 변이 추정이 필요한데 이를 수행하는 알고리즘으로 여러 가지 비디오 코딩 표준들에서 블록 정합 알고리즘을 사용한다. 블록 정합 알고리즘 중 완전탐색 알고리즘은 기준 블록을 탐색영역 안에 존재하는 모든 블록과 비교하여 최적의 블록을 찾아낸다. 이 알고리즘은 최적의 블록을 찾을 수 있어 효율은 좋으나 많은 연산량이 단점이 된다. 본 논문에서는 스테레오 영상에서 움직임 벡터 정보와 전 프레임의 변이 벡터 정보를 이용하여 고속으로 현재 프레임의 변이 벡터를 추정할 수 있는 방안을 제시한다. 변이 벡터 추정시 전역 변이 벡터를 사용하여 탐색 영역을 줄이고, 전 프레임들 사이에서 구한 변이 벡터 정보를 재사용하면서 움직임 벡터 정보를 이용하여 탐색 위치를 제한함으로 연산량을 줄여 고속의 변이 벡터 추정을 가능하게 하였다. 실험결과 제안 알고리즘은 움직임이 많은 복잡 영상 보다는 움직임이 적은 단순 영상에서의 성능이 훨씬 뛰어났으며, 움직임이 적은 단순 영상에서의 변이 벡터 추정 시에 약간의 residual 증가는 있지만 빠른 처리 속도를 제공하여 고속의 변이 벡터 추정을 가능하게 함을 확인하였다.

Abstract

Stereoscopic images consist of the left image and the right image. Thus, stereoscopic images have much amounts of data than single image. Then an efficient image compression technique is needed, the DPCM- based predicted coding compression technique is used in most video coding standards. Motion and disparity estimation are needed to realize the predicted coding compression technique. Their performing algorithm is block matching algorithm used in most video coding standards. Full search algorithm is a base algorithm of block matching algorithm which finds an optimal block to compare the base block with every other block in the search area. This algorithm presents the best efficiency for finding optimal blocks, but it has very large computational loads. In this paper, we have proposed fast disparity estimation algorithm using motion and disparity vector information of the prior frame in stereo image coding. We can realize fast disparity vector estimation in order to reduce search area by taking advantage of global disparity vector and to decrease computational loads by limiting search points using motion vectors and disparity vectors of prior frame. Experimental results show that the proposed algorithm has better performance in the simple image sequence than complex image sequence. We conclude that the fast disparity vector estimation is possible in simple image sequences by reducing computational complexities.

Keywords : disparity vector estimation, stereo image coding, motion vector estimation

* 학생회원, ** 정회원, 중앙대학교 영상공학과
(Graduate School of Advanced Imaging Science, Multimedia & Film, Chung-Ang University)

※ 본 연구는 ITRC(Information Technology Research Center)와 서울시 산학연 협력사업의 지원으로 수행되었음

접수일자: 2009년1월25일, 수정완료일: 2009년8월27일

Ⅰ. 서 론

멀티미디어 콘텐츠의 홍수 속에서 콘텐츠를 사용하는 사용자들의 눈높이가 점점 높아지고 요구사항의 수

준도 높아지면서 실감영상에 대한 관심 또한 높아지고 있다. 실감영상이란 3차원 입체감이 좀 더 사실적으로 느껴지는 영상을 일컫는데, 최근에는 이러한 실감영상을 위한 기술로 양안시점 입체영상 (stereoscopic image), 다시점 비디오 (multi-view video), 홀로그래피 (holography) 등과 같은 3차원 영상과 비디오 기술에 대한 연구가 활발히 진행되고 있다[1].

실감영상을 표현하기 위한 가장 기본적인 방법은 현실세계를 스테레오 영상으로 나타내고, 이 영상 데이터를 처리하여 저장하고 전송하는 것이다. 스테레오 영상은 오른쪽과 왼쪽 2개의 영상만으로 3차원 영상을 만들어 낼 수 있어 많이 사용되고 있다. 그러나 스테레오 영상은 단일영상과는 달리 2개의 영상인 오른쪽과 왼쪽 영상으로 구성되어 있기 때문에 단일영상과 비교하여 더욱 많은 데이터량을 갖게 된다. 따라서 이를 효율적으로 처리하기 위한 영상 압축 기술이 필요하게 되었다.

스테레오 영상은 촬영 시 동일한 대상을 거리가 떨어져 있는 카메라로 동일한 시간에 촬영하기 때문에 앞 영상과 뒤 영상 간, 오른쪽 영상과 왼쪽 영상 간에 중복된 정보들이 많이 존재하게 된다. 이 중복된 정보들을 제거함으로써 영상 데이터의 압축 효과를 높일 수 있다[2]. 중복된 정보를 제거하는 방법으로 앞 영상과 뒤 영상 사이에서는 단일영상 처리에서와 마찬가지로 움직임 추정 방법을 사용하고, 오른쪽 영상과 왼쪽 영상 사이에서는 변이 추정 방법을 사용한다.

움직임 추정과 변이 추정을 수행하는 알고리즘으로 여러 가지 비디오 코딩 표준들에서 블록 정합 알고리즘을 채택하고 있다. 블록정합 알고리즘은 기준이 되는 영상을 정하여 블록단위로 나누고 나머지 영상에서 탐색영역을 기준으로 가장 비슷한 블록을 찾아내어 찾은 블록과 기준 블록의 상대적인 위치 차이인 움직임 벡터와 변이 벡터를 구하고 기준 영상과 구해진 벡터들을 이용해 예측영상을 구성한 후 나머지 영상과의 차이값과 구해진 벡터들만을 전송하는 방법이다.

블록 정합 알고리즘 중 완전탐색 알고리즘은 기준 블록을 탐색영역 안에 존재하는 모든 블록과 비교하여 최적의 블록을 찾아낸다. 이 알고리즘은 최적의 블록을 찾을 수 있어 효율은 좋으나 많은 연산량이 단점이 된다.

본 논문에서는 변이 추정을 위해 움직임 벡터와 전 프레임 사이에서 구해진 변이 벡터를 이용하여 탐색위치를 제한하고, 전역 변이 벡터를 이용하여 탐색영역을 감소시켜 연산량을 줄이는 알고리즘을 제시하고, 움직임 벡터 추정을 위해선 Yih-Chuan Lin와 Shen-Chuan Tai가 제안한 Fast Full-Search Block-Matching Algorithm을[3] 사용하여 완전 탐색과 비교해 탐색결과는 거의 동일하지만 연산량을 줄이는 방법을 이용해, 전체적인 추정과정의 연산량을 감소시키는 방안을 제안한다.

본 논문의 구성을 다음과 같다. Ⅱ장에서는 스테레오 영상 코딩 개요와 Fast Full-Search Block-Matching Algorithm을 소개하고, Ⅲ장에서는 움직임 벡터를 이용한 고속 변이 벡터 추정 알고리즘을 제안하고, ⅣV장에서는 제시하는 방안에 대한 실험 결과를 논한 뒤, V장에서 본 논문의 결론을 맺는다.

Ⅱ. 스테레오 코딩 및 Fast Full-Search Block-Matching Algorithm

1. 스테레오 영상 코딩 개요

스테레오 영상은 촬영시 동일한 대상을 거리가 떨어져 있는 카메라로 동일한 시간에 촬영하기 때문에 앞 영상과 뒤 영상 간, 오른쪽 영상과 왼쪽 영상간에 중복된 정보들이 많이 존재하게 된다. 스테레오 코딩에서는 이러한 영상 데이터에 존재하는 중복된 정보를 제거함으로 영상정보의 압축을 수행한다[2]. 중복된 정보를 제거하는 방법으로 DPCM기반의 예측부호화를 사용한다. 예측부호화의 구현을 위해선 앞, 뒤 영상 사이에서는 움직임 추정을, 오른쪽과 왼쪽 영상 사이에서는 변이 추정을 수행한다. 그림 1에서는 전체적인 스테레오 영상 압축기의 구조를 나타낸다. 왼쪽 영상은 단일 영상 코딩을 구현하기 위해 움직임 추정 방법을 사용하고, 오른쪽 영상은 왼쪽 영상을 같이 이용하여 변이 추정방법을 사용한다.

변이란 같은 객체가 오른쪽 영상과 왼쪽 영상에 존재

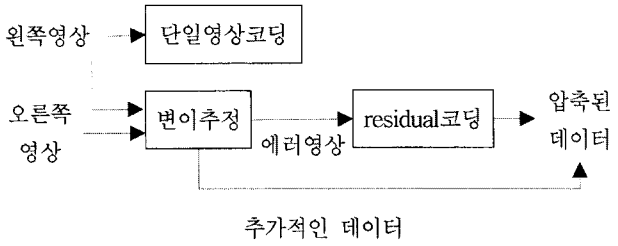

그림 1. 스테레오 영상 압축기 구조
Fig. 1. The structure of stereo image compressor.

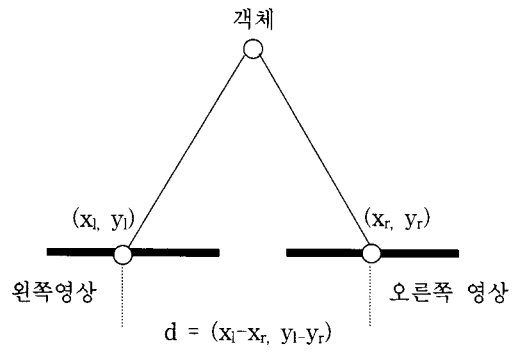

그림 2. 변이
Fig. 2. Disparity.

하는 경우 객체가 위치하는 오른쪽 영상과 왼쪽 영상 사이의 거리로, 오른쪽 영상과 왼쪽 영상에 존재하는 객체의 위치를 (x_r, y_r)과 (x_l, y_l)이라고 하면, 변이 $d=(x_l-x_r, y_l-y_r)$으로 두 위치간의 차이를 나타낸다.[4]

변이 추정 과정에서는 보통 블록 정합 알고리즘을 사용한다. 블록 정합 알고리즘 중 완전 탐색 알고리즘은 기준 블록을 탐색 영역 안에 존재하는 모든 블록과 비교하여 최적의 블록을 찾아내기 때문에 효율은 좋으나 많은 연산량이 단점이 된다.

이러한 단점을 극복하기 위한 빠른 변이 추정 기술들이 연구되어 왔다. 오른쪽 영상과 왼쪽영상의 차영상 정보를 이용하여 추정할 블록의 특성을 정의해 탐색영역을 줄이는 방법[5], 매크로 블록의 예측벡터에 대한 신뢰도를 고려하여 탐색 영역을 조절하는 방법[6], 에피폴라 기하학을 이용하여 탐색 영역을 줄이는 방법[7], 계층구조를 이용한 방법[8], 2단계 가중치 윈도우 알고리즘을 이용하여 탐색 영역을 줄이는 방법[9], 변이 벡터에 관련된 2가지 기하학적 성질을 이용하여 탐색 영역을 줄이는 방법[10] 등 여러 가지 방면으로 연구가 진행되어 왔다.

본 논문에서는 기존의 방법과 비교하여 탐색 영역뿐만 아니라 탐색 위치도 제한하는 방법으로 움직임 벡터 정보를 이용하여 탐색 위치를 제한하고, 전역 변이 벡터를 이용하여 탐색 영역을 감소시켜 기존의 방법보다 연산량을 줄여 고속으로 변이 벡터 추정이 가능한 방안을 제안한다.

2. Fast Full-Search Block-Matching Algorithm (FFBMA)

움직임 벡터 추정 시에도 완전 탐색의 단점을 개선할 많은 방법들이 제안되었다. Three Step Search[11], New Three Step Search[12], Four Step Search[13], Diamond Search[14], Hexagon-Based Search[15] 등등이 있는데 이들 방법은 속도는 빠르나 local minimum에 빠질 가능성이 많아 효율이 좋지 않을 가능성이 많다. 따라서 본 논문에서는 효율은 완전탐색과 비슷하고 상대적으로 속도는 빠른 Fast Full-Search Block-Matching Algorithm[3]을 움직임 벡터 추정시에 사용한다.

Yih-Chuan Lin와 Shen-Chuan Tai가 제안한 Fast Full-Search Block-Matching Algorithm[3]의 기본 개념은 블록 매칭을 수행하는 동안 기준 블록과 후보 블록에 대하여 식 (1)에서 정의된 완전탐색에서 사용하는 매칭 에러를 구하기 전에 3가지의 매칭 기준을 적용하여 기준 블록과 후보 블록의 블록 매칭을 수행할지를 결정하여 움직임 벡터 추정과정의 연산량을 감소시키는 알고리즘이다.

$$D_p(i,j)=\sum_{x=0}^{N-1}\sum_{y=0}^{N-1}|f_t(l+x,k+y)-f_{t-1}(l+i+x,k+j+y)|^p, \quad p=1\text{or}2 \ \text{ and } \ -W\le i,j\le W \tag{1}$$

여기서 $D_p(i,j)$은 매칭 에러 함수이고, N은 블록 크기, W는 탐색영역 크기, $f_t(l,k)$는 좌표 (l,k)에서의 기준 블록의 픽셀값, $f_{t-1}(l+i,k+j)$는 좌표 (l+i,k+j)에서의 후보 블록의 픽셀값이다.

3가지의 매칭 기준은 integral projection으로부터 유도된다. 아래의 식 (2)~(4)는 프레임 t에서 주어진 블록 $f_t(l,k)$에 대한 3가지의 integral projection을 정의한다.

1) Vertical projection

$$v_{t,(l,k)}(y)=\sum_{x=0}^{N-1}f_t(l+x,k+y), \qquad 0\le y\le N-1 \tag{2}$$

2) Horizontal projection

$$h_{t,(l,k)}(x)=\sum_{y=0}^{N-1}f_t(l+x,k+y), \qquad 0\le x\le N-1 \tag{3}$$

3) Massive projection

$$m_{t,(l,k)}=\sum_{x=0}^{N-1}\sum_{y=0}^{N-1}f_t(l+x,k+y) \tag{4}$$

FFBMA에서 제안하는 3개의 빠른 매칭 에러 측정법은 아래 식 (5)~(7)에 정의된다.

$$M_p(i,j) = |m_{t,(l,k)} - m_{t-1,(l+i,k+j)}|^p, \quad -W \le i,j \le W \tag{5}$$

$$V_p(i,j) = \sum_{y=0}^{N-1} |v_{t,(l,k)}(y) - v_{t-1,(l+i,k+j)}(y)|^p, \quad -W \le i,j \le W \tag{6}$$

$$H_p(i,j) = \sum_{x=0}^{N-1} |h_{t,(l,k)}(x) - h_{t-1,(l+i,k+j)}(x)|^p, \quad -W \le i,j \le W \tag{7}$$

처음에 좌표 (0, 0)에서의 움직임 벡터 (i*, j*)를 최적의 움직임 벡터라고 가정하고, 여기에서 계산된 MSE를 D_2(i*, j*)라고 한 후 탐색 영역 내에서 다른 후보 블록들을 검사할 때 아래에 제시한 4가지 조건 중 1), 2), 3) 조건을 적용하여 조건이 하나라도 맞을 경우 후보 블록에 대한 수행을 중지하고 다음 후보 블록의 검사를 수행하고, 3가지 조건에 모두 맞지 않으면 완전탐색을 수행하고 4) 조건에 맞지 않으면 초기에 설정한 움직임 벡터와 MSE 값을 현재 구한 벡터와 MSE값으로 대체함으로 최적의 움직임 벡터를 찾아낸다.

1) $N^2 D_2(i_*, j_*) < M_2(i,j)$

2) $ND_2(i_*, j_*) < V_2(i,j)$

3) $ND_2(i_*, j_*) < H_2(i,j)$

4) $D_2(i_*, j_*) < D_2(i,j)$

조건에서 사용할 integral projection들의 값을 계산할 때 2개의 후보 블록인 $f_{t-1}(l,k-1)$와 $f_{t-1}(l-1,k)$의 값을 알면 대입과 간단한 덧셈과 뺄셈의 계산만으로 $f_{t-1}(l,k)$ 블록의 integral projection을 구할 수 있어 연산량을 감소시키게 된다.

Ⅲ. 움직임 벡터를 이용한 고속 변이 벡터 추정 알고리즘

스테레오 영상들 사이에는 시간적, 공간적인 상관관계가 존재한다. 그림 3의 Loop constraint는 2개의 연속되는 시간에 위치한 2개의 영상에 대한 움직임 벡터와 변이 벡터 사이의 관계를 나타낸다[16].

$$\overrightarrow{M_l} + \overrightarrow{D_t} - \overrightarrow{M_r} - \overrightarrow{D_{t+1}} \approx \vec{0} \tag{8}$$

여기서 $\overrightarrow{M_l}$과 $\overrightarrow{M_r}$은 왼쪽 영상과 오른쪽 영상의 움직임 벡터이고, $\overrightarrow{D_t}$와 $\overrightarrow{D_{t+1}}$은 t 프레임과 t+1 프레임에서의 스테레오 영상의 변이 벡터이다. 이 관계를 살펴보면 $\overrightarrow{M_l}$ 이나 $\overrightarrow{D_t}$ 중 어느 하나라도 값이 변하면 $\overrightarrow{M_r}$이나 $\overrightarrow{D_{t+1}}$의 값도 같이 변화 된다는 것을 알 수 있다. 이 점에 착안하여 본 논문에서의 첫번째 제안 방법인 움직임 벡터 정보를 이용하여 변이 벡터 탐색 범위를 제한하는 방안을 다음과 같이 제시한다.

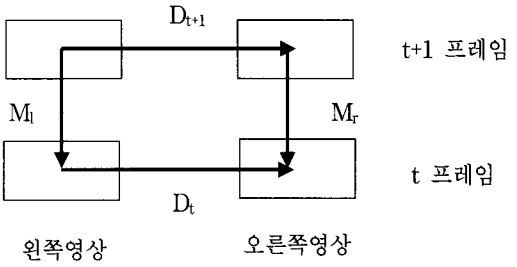

그림 3. 순환 제약 조건
Fig. 3. Loop Constraint.

어떤 위치의 움직임 벡터 값이 존재한다는 것은 그 위치에 존재하는 객체가 움직이거나 카메라가 움직인 경우라고 할 수 있고, 움직임 벡터 값이 0 이라면 움직임이 없거나 배경이라고 간주할 수 있다. 이것은 오른쪽 영상이나 왼쪽 영상 동일하게 적용되며, 움직임 벡터의 값이 0 인 경우, 즉 $\overrightarrow{M_l}$ = $\overrightarrow{M_r}$ = 0 인 경우 식 (8)은 $\overrightarrow{D_t} \approx \overrightarrow{D_{t+1}}$가 되어 $\overrightarrow{M_l}$ = 0 이면 t+1 프레임에서의 오른쪽 영상과 왼쪽 영상 사이에서의 변이 벡터 $\overrightarrow{D_{t+1}}$는 추정 과정을 거치지 않고 t 프레임에서 구한 $\overrightarrow{D_t}$을 그대로 사용할 수 있다. 이 관계를 아래와 같이 나타낼 수 있다.

$$IF\ \overrightarrow{M_l} == 0\ THEN\ \overrightarrow{D_{t+1}} = \overrightarrow{D_t}$$
$$ELSE\ \overrightarrow{D_{t+1}} = New(\overrightarrow{D_{t+1}})$$

t 프레임에서의 스테레오 영상의 변이 벡터 $\overrightarrow{D_t}$와 왼쪽 영상에서의 t 프레임과 t+1 프레임 사이의 움직임 벡터 $\overrightarrow{M_l}$이 구해져 있다고 가정하고, t+1 프레임에서의 스테레오 영상의 변이 벡터 $\overrightarrow{D_{t+1}}$을 추정할 때 $\overrightarrow{M_l}$의 값을 조사하여 $\overrightarrow{M_l}$의 값이 존재하는 블록에 대해서는 영상내의 객체의 움직임이 있다고 판단하여 $\overrightarrow{D_{t+1}}$을 새롭게 구하고, $\overrightarrow{M_l}$의 값이 존재하지 않는 블록에 대해서는 영상내의 객체의 움직임이 없거나 배경으로 판단

하여 $\overrightarrow{D_t}$의 값을 그대로 사용해 변이 벡터 추정과정을 생략하는 방법으로 탐색 위치를 제한하여 변이 벡터 추정 속도를 향상시켰다.

본 논문에서 제안하는 두 번째 방법으로 탐색 영역을 줄이기 위하여 전역 변이 벡터를 사용한다. 전역 변이 벡터는 프레임 전체에서 구한 변이 벡터들의 평균으로 구한다[18].

$$GDV = \sum_{i=1}^{N} \overrightarrow{D_t(i)} / N, \quad N = count(\overrightarrow{D_t}) \tag{9}$$

여기서 GDV는 전역 변이 벡터를 나타내고 N는 프레임 전체에서 구한 변이 벡터들의 갯수를 나타낸다. 구해진 전역 변이 벡터를 이용하여 탐색 영역을 y축으로는 전역 변이 벡터 y값 ~ -전역 변이 벡터 y값 사이로 하고 x축으로 전역 변이 벡터 x값 ~ (전역 변이 벡터 x값 - 매크로 블록 크기)로 정하여 탐색 영역을 축소시켜 변이 벡터 추정시 연산량을 감소시켰다

t 프레임에서 구한 변이 벡터 $\overrightarrow{D_t}$를 t+1 프레임에서 $\overrightarrow{D_{t+1}}$를 구하는데 그대로 사용하는 과정에서 에러 전파가 발생한다. 이를 보정하기 위해서 residual의 크기를 문턱치(TH_{res})로 잡아 문턱치 이상이 되면 t+1 프레임에서 전체적으로 $\overrightarrow{D_{t+1}}$을 다시 구해 변이 벡터의 에러를 수정하고 residual의 크기도 줄이는 과정을 거쳐 에러를 보정하고 전역 변이 벡터를 다시 구한다. residual은 나머지 영상과 예측 영상의 차이값으로 원래의 영상 데이터보다 데이터 값이 작아 사용하는 비트를 줄이기 때문에 영상 압축 처리에선 residual을 코딩하여 전송함으로써 압축을 수행한다. 문턱치를 낮게 잡으면 프레임 전체적으로 변이 벡터를 구하는 횟수가 증가하여 연산량이 늘어 처리 속도가 증가 하게 되나 residual은 작아지고, 문턱치를 높게 잡으면 처리 속도는 감소하나 residual을 증가하게 되어 처리 속도와 residual간의 트레이드 오프 관계가 존재하여 원하는 결과의 문턱치를 잡아 속도를 빠르게 하거나 residual의 크기를 작게 하는 조절이 가능하다.

전체 처리 시스템은 크게 입력 영상들에 대한 전처리 부분, 첫번째 프레임 및 residual의 크기 > TH_{res}인 프레임에 대한 처리로 프레임 전체에 대한 변이 벡터와 전역 변이 벡터를 구하는 부분, 그 외의 프레임 처리로 t-1 프레임에서 구한 변이 벡터 $\overrightarrow{D_{t-1}}$와 움직임 벡터 $\overrightarrow{M_t}$을 이용하여 변이 벡터를 구하는 부분, 이렇게 3가지 부분으로 구성된다. 처리하는 영상에 대한 조건으로는 오른쪽 영상과 왼쪽 영상이 수평으로 정렬되어 있다고 가정한다.

1. 입력 영상들에 대한 전처리

변이 추정 시 에러함수로 SAD를 사용하기 때문에 변이 추정을 수행하기 전 스테레오 영상은 반드시 image balance작업을 필요로 한다. 같은 종류의 카메라로 영상을 촬영한다고 해도 카메라의 위치가 다르기 때문에 빛의 효과가 다르고 카메라의 물리적인 미묘한 차이 등으로 해서 각각의 카메라가 받아들이는 영상정보엔 차이가 발생할 가능성이 많다. 이 차이를 평준화 하는 것이 image balance인데 이 작업이 이루어 지지 않으면 변이 추정 시 오류가 많이 발생하여 제대로 된 변이 추정이나 움직임 추정을 어렵게 한다. 수행방법은 식 (10)에 정의되어 있다[16].

$$\hat{I}_R(i,j) = \frac{\sigma_L}{\sigma_R} I_R(i,j) + (m_L - \frac{\sigma_L}{\sigma_R} m_R) \tag{10}$$

여기서 σ_L과 σ_R은 왼쪽 영상과 오른쪽 영상의 표준편차이고, m_L과 m_R은 평균값, $I_R(i,j)$은 오른쪽 영상의 값이고 $\hat{I}_R(i,j)$은 image balance 수행 후의 값이다.

본 논문에서는 변이 벡터를 추정하기 위해 특징점을 추출하는데 특징점을 추출하는 방법으로 Sobel Edge Detector를 사용한다. Sobel Edge Detector를 사용하기 위해선 처리하는 영상에 대한 문턱치 값을 정해야 하는데 그 방법으로 Iterator selection histogram을 이용하여 전처리 단계에서 영상에 대한 문턱치(TH_{sob})값을 결정 한다[17].

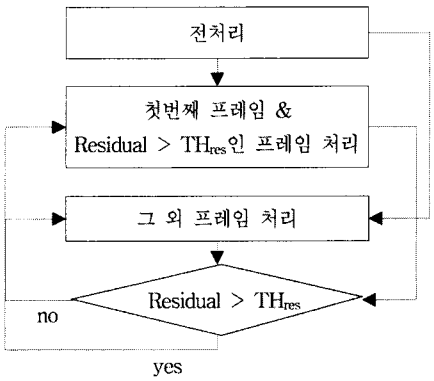

그림 4. 제안 시스템 구성도
Fig. 4. The structure of proposed system.

본 논문에서는 2단계에서의 변이 벡터 추정 시 탐색 영역을 줄이기 위해 에피폴라 라인을 사용한다. 에피폴라 라인을 구하기 위해선 카메라의 intrinsic matrix와 extrinsic matrix를 이용하여 fundamental matrix를 구하는데 이 처리 또한 전처리 단계에서 수행한다.

2. 첫번째 프레임 및 residual이 문턱치 보다 높아졌을 경우에 해당하는 프레임 처리

이 단계의 목적은 프레임 전체적으로 변이 벡터를 구하고, 전역 변이 벡터의 크기를 정하는 것이다. 전역 변이 벡터는 프레임 전체에서 구한 변이 벡터들의 평균[18]으로 본 논문에서는 3단계에서 t+1 프레임에서 변이 벡터들을 구할 때 탐색 영역을 축소시키기 위해서 사용한다. 프레임 전체적으로 변이 벡터를 구하는 방법은 완전 탐색과 거의 유사하며 구해진 변이 벡터와 residual의 크기는 완전 탐색의 결과와 동일하기 때문에 3단계 처리 시 residual의 값이 TH_{res}보다 높아진 경우 다시 2단계를 수행함으로 3단계에서 발생하는 에러전파를 보정하게 된다. 처리과정은 다음과 같다.

Step 1 : 변이 벡터를 매크로 블록 단위로 1개씩 찾아내기 때문에 왼쪽 영상을 매크로 블록 크기로 나눈다.

Step 2 : 왼쪽 영상의 매크로 블록 단위 기준으로 오른쪽 영상에서 1단계에서 구한 fundamental matrix를 이용하여 에피폴라 라인을 구한다. 에피폴라 라인은 y축 방향의 탐색 영역을 줄이는데 사용한다. 영상이 수평으로 정렬되어 있기 때문에 이 Step의 수행은 수평방향으로 처음에 한번씩만 수행한다.

Step 3 : 왼쪽 영상의 매크로 블록에 대해 Sobel edge detector를 이용하여 특징점을 구한다. 특징점이 검사한 매크로 블록에 존재하지 않으면 다음 블록에 대하여 특징점이 존재하는지 검사하고, 특징점이 존재하면 Step4를 수행한다. Sobel edge detector를 이용할 때 1단계에서 구한 영상의 문턱치(TH_{sob})를 사용한다.

Step 4 : 왼쪽 영상의 매크로 블록에 대해 오른쪽 영상에서의 탐색 영역내에 존재하는 모든 블록을 비교하여 SAD가 가장 작은 블록을 찾아내어 변이 벡터를 결정하고, 전역 변이 벡터를 구하기 위해 변이 벡터의 x축 방향 값과 y축 방향 값을 합산한다. 에러 함수로 사용하는 SAD에 대한 수식은 식 (1)에 정의되어 있다.

Step 5 : 수평 방향으로 마지막 블록이면 Step2로 가서 다음 블록 라인의 수평 방향으로 에피폴라 라인을 구하고, 프레임의 마지막 블록이면 전역 변이 벡터를 구한다.

3. 그 외 프레임 처리 방법

이 단계가 제안 알고리즘의 핵심으로 t-1 프레임에서 구한 변이 벡터 $\overrightarrow{D_{t-1}}$와 움직임 벡터 $\overrightarrow{M_l}$을 이용하여 t 프레임에서 빠르게 변이 벡터 $\overrightarrow{D_t}$를 구하는 부분이다.

움직임 벡터의 크기가 0이면 t-1 프레임에서 구한 변이 벡터를 그대로 사용하고 0이 아니면 $\overrightarrow{D_t}$를 다시 구함으로 변이 벡터 추정시 탐색 위치를 제한하여 변이 벡터 추정 속도를 향상시켰다. 또한 3단계에서는 에피폴라 라인을 사용하지 않고 2단계에서 구한 전역 변이 벡터를 다음과 같이 이용하여 탐색 영역을 축소시켜 변이 벡터 추정시 연산량을 감소시켰다.

$$-GDY \leq y \leq GDY$$

$$GDX - macroblocksize \leq x \leq GDX$$

여기서 GDY는 전역 변이 벡터의 y 값이고 GDX는 전역 변이 벡터의 x값이고 x와 y는 탐색영역의 범위이다. 처리과정은 다음과 같다.

Step 1 : 변이 벡터를 매크로 블록 단위로 1개씩 찾아내기 때문에 왼쪽 영상을 매크로 블록 크기로 나눈다.

Step 2 : 왼쪽 영상의 t 프레임과 t-1 프레임 사이에서 구한 움직임 벡터 값을 검사한다. 움직임 벡터 값이 0이면 t-1 프레임에서 구한 변이 벡터 $\overrightarrow{D_{t-1}}(i,j)$의 값을 $\overrightarrow{D_t}(i,j)$에서 재사용하고 다음 블록의 움직임 벡터를 검사한다. 움직임 벡터의 값이 0이 아니면 Step 3을 수행한다.

Step 3 : 왼쪽 영상에서 움직임 벡터 값이 0이 아닌 블록에 대해 Sobel edge detector를 이용하여 특징점을 구한다. 특징점이 검사한 매크로 블록에 존재하지 않으면 Step 2로 가서 다음 블록의 움직임 벡터 값을 검사하고, 특징점이 존재하면 Step4를 수행한다. 여기서도 1단계에서 구한 영상의 문턱치(TH_{sob})를 Sobel edge detector에 사용한다.

Step 4 : 왼쪽 영상의 매크로 블록에 대해 오른쪽 영상에서의 탐색 영역내에 존재하는 모든 블록을 비교하여 SAD가 가장 작은 블록을 찾아내어 변이 벡터를 결정한다.

Ⅳ. 실험 및 평가

제안한 알고리즘의 성능을 측정하기 위하여 실험은 MATLABR2007을 사용하여 수행하였고, 실험 영상은 MERL에서 제공하는 640X480 크기의 스테레오 동영상인 VASSAR 영상, EXIT 영상과 BALLROOM 영상 250 프레임을 사용하였다.

변이 벡터를 구하는 알고리즘의 성능을 비교, 평가하기 위해서 완전 탐색과 Fast algorithm[6] 및 제안 알고리즘을 구현하여 연산량과 residual의 크기를 비교하였다. 연상량은 덧셈, 뺄셈, 곱셈, 나눗셈, 비교, 5가지 연산의 총 연산량을 프레임 단위로 평균을 내어 비교하였다.

변이 벡터를 구하기 위해 Sobel edge detector를 사용하여 특징점을 찾아내고 특징점이 존재하는 블록에 대해서만 탐색 영역 내에서 완전 탐색을 수행하였다.

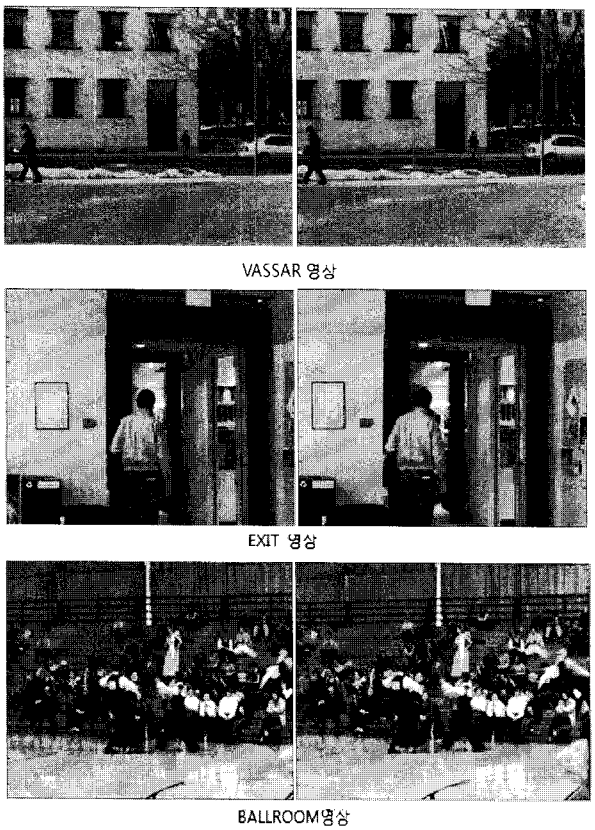

그림 5. 테스트 영상들
Fig. 5. Test seauences.

에러 측정 방법으로는 SAD를 사용하여 SAD가 최소인 블록을 검색하였다. Sobel edge detector에서 사용하는 문턱치 값은 Iterator selection histogram 방법을 이용하여 측정해 VASSAR는 57, EXIT는 67, BALLROOM은 93을 사용하였다.

완전 탐색과 Fast algorithm에 대해서도 제안 알고리즘의 전처리 과정인 image balance 작업과 sobel edge detector에서 사용할 문턱치 값을 결정하는 작업을 동일하게 수행한다. 또한, 완전 탐색과 Fast algorithm에서도 변이 벡터를 구할 때 특징점을 찾아내서 특징점이 존재하는 블록에서만 탐색 영역 내에서 완전 탐색을 수행하였고 최소 SAD인 블록을 검색하였다.

완전 탐색과 Fast algorithm 및 제안 알고리즘의 2단계에선 탐색 영역을 48X4 크기의 탐색 영역을 사용하였고 제안 알고리즘의 3단계에선 16X4 크기의 탐색 영역을 사용하였다. Fast algorithm의 경우엔 처음 시작은 48X4크기로 하지만 Fast algorithm에 따라 탐색영역이 적응적으로 변한다. 제안 알고리즘 3단계에서 사용하는 탐색 영역의 크기가 작아진 이유는 제안 알고리즘 2단계에서 구한 전역 변이 벡터를 사용하여 탐색 영역을 줄였기 때문이다.

특징점을 기반으로 변이 벡터를 구하기 때문에 실험 결과로써 비교하기 위한 residual을 구하기 위해 특징점이 존재하지 않는 블록에 대한 변이 벡터 할당이 필요하여 다음과 같이 수행하였다. 만약 첫번째 블록에 특징점이 존재하지 않아 변이 벡터가 존재하지 않는다면 전역 변이 벡터를 할당하고 그 다음 블록부터는 변이 벡터가 존재하지 않는 블록엔 바로 전 블록의 변이 벡터를 할당하였다. 이 과정은 완전 탐색과 Fast algorithm 및 제안 알고리즘 모두에 동일하게 사용하였다.

본 논문에서는 추정된 변이 벡터와 왼쪽 영상을 이용하여 예측 영상을 만든 후, 예측 영상과 오른쪽 영상과의 차이값을 residual로 구했다. residual 데이터 값 x에 대해 $2^{n-1} < x \leq 2^n$ 범위에 존재하면 n을 residual의 크기로 정하여 전체 프레임에 대한 residual의 크기를 구한 후 비트 당 크기를 구하였다.

제안 알고리즘에서는 문턱치로 사용하는 residual의 값을 완전 탐색과 Fast algorithm의 결과 residual의 값을 참조하여 설정하고 실험을 수행하였으며 각각을 제안(residual크기)로 그림과 표에 표시하였다. 그림 6는 완전 탐색과 Fast algorithm 및 각 문턱치에 해당되

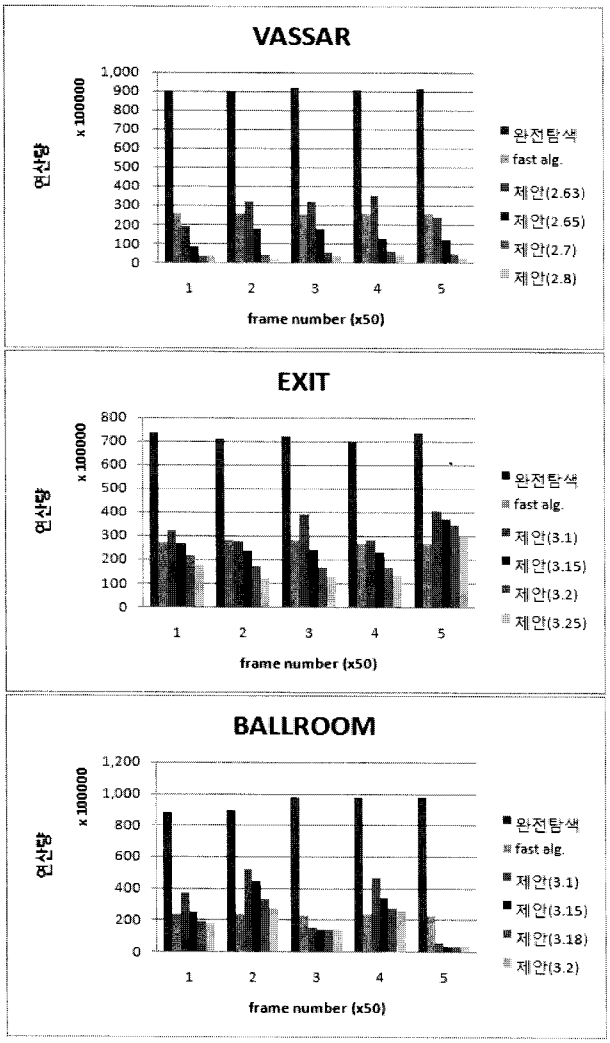

그림 6. 연산량 크기 비교 결과
Fig. 6. Comparison of computational amounts.

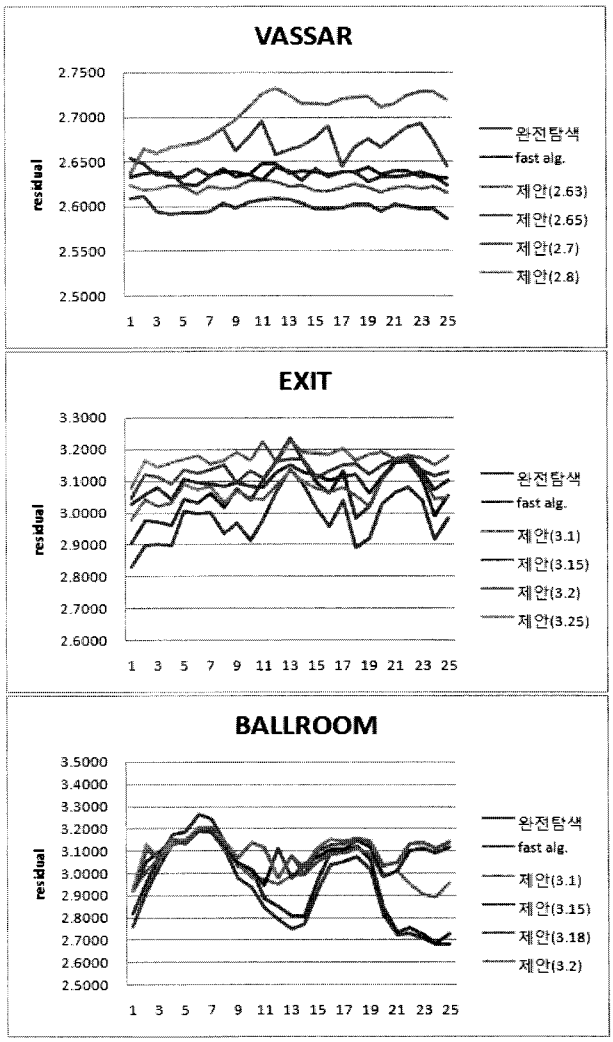

그림 7. Residual 크기 비교 결과
Fig. 7. Comparison of residual size.

는 제안 알고리즘의 연산량 비교 결과 그래프이고 각 영상마다 50프레임 단위로 평균을 나타내었다. 또한, 그림 7은 완전 탐색과 Fast algorithm 및 각 문턱치에 해당되는 제안 알고리즘의 residual 비교 결과 그래프로 각 영상마다 10프레임 단위로 평균을 나타내었고, 표 1에서는 전체 프레임에 대한 변이 벡터 실험 결과 평균을 정리하였다. 표 1에서의 비교 1은 완전탐색과 제안 알고리즘의 비교 결과치이고 비교2는 Fast algorithm과 제안 알고리즘의 비교 결과치이다.

실험결과를 살펴보면 움직임이 많고 복잡한 영상인 BALLROOM 영상에서는 완전탐색과 비교하여 최대 82%의 연산량이 감소하였지만 residual이 6.4% 증가한 것을 보여주고 있고, Fast algorithm과 비교해서는 최대 25%의 연산량 감소와 4.8%의 residual증가를 보여주고 있다. BALLROOM 영상보다 움직임이 적고 복잡도가 낮은 영상인 EXIT의 경우 완전탐색과 비교하여 최대 연산량이 77% 감소하고 residual은 6.4% 증가하였고, Fast algorithm과 비교해서는 최대 39%의 연산량 감소와 3.5%의 residual증가를 보여줘 BALLROOM 영상의 실험결과와 비교하면 완전탐색과 비교한 결과에서는 연산량 면에서 성능이 떨어지나 Fast algorithm과 비교한 결과에서는 연산량과 residual 모두 더 좋은 성능을 보여주고 있다. 3가지의 테스트 영상중 가장 움직임이 적고 단순한 영상인 VASSAR 영상의 결과를 보면 완전탐색과 비교하여 최대 97%의 연산량 감소를, 3.9%의 residual증가를 보여주고 있고, Fast algorithm과 비교하여 최대 89%의 연산량이 감소하고 2.4%의 residual이 증가해 3가지 영상중 가장 뛰어난 성능을 보여주고 있다. 따라서 본 논문의 제안 알고리즘은 움직임이 적은 단순 영상에서의 변이 벡터 추정 시에 약간의 residual

표 1. 변이 벡터 실험 결과
Table 1. Results of disparity vector experiment.

동영상	추정방법	연산량	비교1	비교2
VASS AR	완전탐색	90,861,758	-	-
	Fast Alg.	25,526,586	72%감소	-
	제안(2.63)	28,302,523	69%감소	10%증가
	제안(2.65)	13,504,735	86%감소	48%감소
	제안(2.7)	4,349,171	96%감소	83%감소
	제안(2.8)	2,951,080	97%감소	89%감소
	추정방법	**residual**	**비교1**	**비교2**
	완전탐색	2.6001	-	-
	Fast Alg.	2.6373	1.4%증가	-
	제안(2.63)	2.6214	0.8%증가	0.7%감소
	제안(2.65)	2.6356	1.3%증가	0.1%감소
	제안(2.7)	2.6704	2.7%증가	1.2%증가
	제안(2.8)	2.7027	3.9%증가	2.4%증가
EXIT	**추정방법**	**연산량**	**비교1**	**비교2**
	완전탐색	71,819,951	-	-
	Fast Alg.	27,143,867	63%감소	-
	제안(3.1)	33,390,954	54%감소	23%증가
	제안(3.15)	26,817,313	63%감소	2%감소
	제안(3.2)	21,195,845	71%감소	22%감소
	제안(3.25)	16,827,708	77%감소	39%감소
	추정방법	**residual**	**비교1**	**비교2**
	완전탐색	2.9815	-	-
	Fast Alg.	3.0637	2.7%증가	-
	제안(3.1)	3.0730	3%증가	0.3%증가
	제안(3.15)	3.1011	4%증가	1.2%증가
	제안(3.2)	3.1319	5%증가	2.2%증가
	제안(3.25)	3.1729	6.4%증가	3.5%증가
BALL ROOM	**추정방법**	**연산량**	**비교1**	**비교2**
	완전탐색	93,742,207	-	-
	Fast Alg.	22,787,933	76%감소	-
	제안(3.1)	31,163,944	67%감소	36%증가
	제안(3.15)	23,900,721	75%감소	4%증가
	제안(3.18)	18,990,966	80%감소	17%감소
	제안(3.2)	17,103,421	82%감소	25%감소
	추정방법	**residual**	**비교1**	**비교2**
	완전탐색	2.9185	-	-
	Fast Alg.	2.9617	1.4%증가	-
	제안(3.1)	3.0335	3.9%증가	2.4%증가
	제안(3.15)	3.0793	5.5%증가	3.9%증가
	제안(3.18)	3.1011	6.2%증가	4.7%증가
	제안(3.2)	3.1067	6.4%증가	4.8%증가

증가는 있지만 빠른 처리 속도를 제공하여 고속의 변이 벡터 추정을 가능하게 함을 확인하였다.

V. 결론 및 향후 연구과제

본 논문에서는 스테레오 영상에서 움직임 벡터와 이전 프레임 간의 변이 벡터를 이용하여 고속의 변이 벡터 추정 알고리즘을 제안 하였다. 이전 프레임간의 변이 벡터를 재사용하고 움직임 벡터 정보를 이용하여 움직임 벡터가 존재하는 블록에 대해서만 다시 변이 벡터를 구하여 탐색 위치를 제한하고, 전역 변이 벡터를 이용하여 탐색 영역을 감소시켜 변이 추정에 필요한 연산량을 감소시켰다.

실험 결과 본 논문의 제안 알고리즘은 움직임이 많고 복잡한 영상보다는 움직임이 적고 단순한 영상에서의 성능이 뛰어남을 보여주었고, 움직임이 적은 단순 영상에서의 변이 벡터 추정 시에 약간의 residual증가는 있지만 빠른 처리 속도를 제공하여 고속의 변이 벡터 추정을 가능하게 함을 확인하였다.

향후 변이 벡터 추정 시 연산량은 제안 알고리즘의 정도를 유지하면서 추정의 정확도를 높여 현재 결과로 나타난 residual의 크기를 보다 줄여 압축의 효율을 향상시키는 연구가 필요 할 것이다.

참 고 문 헌

[1] 호요성, 오관정, "다시점 비디오 부호화 기술 동향", 전자공학회지, 제34권, 제8호, 8. 2007.

[2] A. Tamtauoui and C. Labit, "Constrained disparity and motion estimators for 3DTV image sequence coding," Signal Processing : Image Comm., Vol. 4, pp. 45-54, Nov. 1991.

[3] Y.-C. Lin and S.-C. Tai, "Fast Full-Search Block-Matching Algorithm for Motion-Compensated Video Compression", IEEE Transactions on Communications, vol. 45, no. 5, pp. 527-531, May 1997.

[4] Tamas Frajka, Kenneth Zeger, "Residual image coding for stereo image compression", Optical Engineering, vol.42, No.1, Jan. 2003.

[5] 박상현, 이상호, 김미현, 손광훈, "차영상 정보를 이용한 효율적인 3차원 영상 변이 추정에 관한 연구", 한국방송공학회 학술대회, pp 9-12, 12. 1998.

[6] Yongtae Kim, Jiyoung Kim, Kwanghoon Shon, "Fast Disparity and Motion Estimation for Multi-view Video Coding", IEEE Transactions on Consumer Electronics, vol.53, No.2, May. 2007.

[7] Jiangbo Lu, Hua Cai, Jian-Guang Lou, Jiang Li, "An Epipolar Geometry-Based Fast Disparity Estimation Algorithm for Multiview Image and Video Coding", IEEE Transactions on Circuits and Systems for Video Technology, vol.17, No.6, Jun. 2007.

[8] 안재균, 김창수, "실시간 스테레오 동영상 처리를

위한 빠른 디스패리티 추정", 대한전자공학회 하계 종합학술대회, 제30권, 제1호, 2007.
[9] Jiyonh Park, Wonjae Lee, Jaeseok Kim, "New Fast Disparity Estimation Algorithm by Diminishing Search Range and FPGA Implementation", IEEE International Symposium on Consumer Electronics, volume. Issu, sept.1-3, 2004. Pp 457-460
[10] Won-Ho Kim, Sung-Woong Ra, "Fast Disparity Estimation Using Geometric Properties and Selective Sample Decimation for Stereoscopic Image Coding", IEEE Transactions on Comsumer Electronics, vol.45, No.1, FEB. 1999.
[11] Iain E.G Richardson, "H.264 and MPEG-4 Video Compression: Video Coding for Next-genertion Multimedia", Wiley, Dec. 2003.
[12] R. Li, B. Zeng, and M. L. Liou, "A new three-step search algorithm for block motion estimation," IEEE Trans. Circuits Syst. Video Technol.,vol. 4, no: 4, pp. 438-442, Aug. 1994.
[13] L. M. Po and W. C. Ma, "A novel four-step search algorithm for fast block motion stimation," IEEE Transactions on Circuits & systems for Video Technology, vol. 6, pp. 3 13-3 17, June 1996.
[14] Shan Zhu, Kai-Kuang Ma, "A new diamond search algorithm for fast block-matching motion estimation," IEEE Transactions on Image Processing, vol. 9, no. 2, pp. 287-290, Feb. 2000.
[15] Ce Zhu, Xiao Lin, Lap-Pui Chau, Keng-Pang Lim, Hock-Ann Ang, and Choo-Yin Ong, "A novel hexagon-based search algorithm for fast block motion estimation, In Proc. of the IEEE International Conference on Acoustics, Speech, and Signal Processing, pp. 1593-1596, May 2001.
[16] W. Yang, K. Ngan, J. Lim, and K. Sohn, "Joint motion and disparity fields estimation for stereoscopic video sequences," Signal Process.: Image Commun., vol. 20, no. 3, pp. 265 - 276, Mar. 2005.
[17] 이문호, 정성환, "오픈소스 CxImage를 이용한 Visual C++ 디지털 영상처리", 홍릉과학출판사, 8.2006.
[18] Yongtae Kim, Seungchul Choia, Sukhee Cho, Kwanghoon Sohn, "Efficient disparity vector coding for multiview sequences", Signal Process.: Image Commun. vol. 19, pp. 539 - 553, 2004.

저 자 소 개

도 남 금(학생회원)
1991년 단국대학교
화학공학과 학사졸업
1996년 중앙대학교
전자계산학과 학사졸업
1996년~2001년 (주)동양시스템즈
2002년~2006년 (주)코디얼
2009년 중앙대학교 영상공학과 석사졸업
<주관심분야 : 영상 및 신호처리, MPEG/H.264>

김 태 용(정회원)-교신저자
1986년 한양대학교
전기공학과 학사졸업
1988년 한양대학교
전자통신공학과 석사졸업
1999년 포항공과대학교
컴퓨터공학과 박사졸업
2003년~현재 중앙대학교 영상공학과 교수
<주관심분야 : 컴퓨터비젼, 영상통신, 인공지능, 게임>