단위 다이아몬드와 납작한 육각패턴을 이용한 고속 블록 정합 알고리즘

(A Fast Block Matching Algorithm using Unit-Diamond and Flat-Hexagonal Search Patterns)

남 현 우 † 위 영 철 †† 김 하 진 ††
(Hyeon-Woo Nam) (Young-Cheul Wee) (Ha-Jine Kimn)

요 약 서로 다른 형태와 크기를 가지는 탐색패턴과 움직임 벡터의 분포는 블록 정합 알고리즘에서 탐색 속도와 화질을 좌우하는 중요한 요소이다. 본 논문에서는 단위 다이아몬드패턴과 납작한 육각패턴을 이용한 새로운 고속 블록 정합 알고리즘을 제안한다. 이 알고리즘은 단위 다이아몬드패턴을 이용하여 적은 탐색점으로 움직임이 적은 벡터를 우선 찾은 다음에 움직임이 큰 벡터에 대해서는 납작한 육각패턴을 이용하여 고속으로 움직임 벡터를 찾게 하였다.

실험결과, 제안된 알고리즘은 육각패턴 탐색 알고리즘에 비하여 움직임 벡터 예측의 속도에 있어서 약 11~51%의 높은 성능 향상을 보였으며 화질 또한 PSNR 기준으로 약 0.05~0.74dB의 향상을 보였다.

키워드 : 단위 다이아몬드패턴, 납작한 육각패턴, 고속 블록 정합 알고리즘

Abstract In the block matching algorithm, search patterns of different shapes or sizes and the distribution of motion vectors have a large impact on both the searching speed and the image quality. In this paper, we propose a new fast block matching algorithm using the unit-diamond search pattern and the flat-hexagon search pattern. Our algorithm first finds the motion vectors that are close to the center of search window using the unit-diamond search pattern, and then fastly finds the other motion vectors that are not close to the center of search window using the flat-hexagon search pattern.

Through experiments, compared with the hexagon-based search algorithm(HEXBS), the proposed unit-diamond and flat-hexagonal pattern search algorithm(UDFHS) improves as high as 11~51% in terms of average number of search point per motion vector estimation and improves about 0.05~0.74dB in terms of PSNR(Peak Signal to Noise Ratio).

Key words : Unit-Diamond Pattern, Flat-Hexagonal Search Pattern, FBMA(Fast Block Matching Algorithm)

1. 서 론

동영상의 빠른 전송 또는 효율적인 저장을 위해서는 동영상 내에 존재하는 시간적, 공간적 중복성을 동영상 분석 기법을 통해 제거하는 압축이 필요하다. 움직임 예측(ME: motion estimation)은 동영상 프레임간의 움직임 벡터(MV: motion vector)를 찾아 그 위치에 해당되는 이전 프레임의 블록과의 차를 부호화함으로써 시간적 중복성을 감소시켜 압축 효율을 증가시킬 수 있는 중요한 요소이다.

영상으로부터 움직임을 분석하기 위해 전통적으로 많이 사용하는 알고리즘은 크게 화소 단위의 움직임 벡터를 추출하는 화소 재귀적 알고리즘(PRA: pel recursive algorithm)과 블록 단위의 움직임 벡터를 추출하는 블록 정합 알고리즘(BMA: block matching algorithm)이 있다. 화소 재귀적 알고리즘은 정확도가 높고 종류가 다양하지만 영상의 전체 범위에 걸쳐 화소 단위의 복잡한 연산을 수행함으로써 처리 시간이 오래 걸리는 단점이 있다. 따라서 블록 정합 알고리즘이 화소 재귀적 알고리즘보다 널리 사용되고 있으며 대표적인 알고리즘은 전역 탐색(FS: full search)이다. 이 알고리즘은 영상을 모양과 크기가 동일한 사각형 블록으로 분할한 후 정합척

† 정 회 원 : 아주대학교 정보통신대학
manner@ajou.ac.kr
†† 종신회원 : 아주대학교 정보및컴퓨터공학부 교수
ycwee@ajou.ac.kr
hjkimn@ajou.ac.kr
논문접수 : 2003년 8월 12일
심사완료 : 2003년 10월 29일

도를 탐색영역 내의 블록들에 적용하여 움직임 벡터를 찾는다. 전역 탐색은 과정이 간단하고 하드웨어 구현이 용이하며 정합오차가 가장 작은 움직임 벡터를 찾을 수 있지만 많은 계산량이 필요한 단점이 있다. 이러한 전역 탐색의 단점을 극복하기 위해 속도가 개선된 3단계 탐색(TSS: three step search)[1], 새로운 3단계 탐색(NTSS: new three step search)[2], 4단계 탐색(FSS: four step search)[3], 다이아몬드 탐색(DS: diamond search)[4], 중심지향적 하이브리드 탐색(CBHS: center-biased hybrid search)[5], 육각패턴 탐색(HEXBS: hexagon-based pattern search)[6,7], 탄력적인 다이아몬드 탐색(AMSED: adaptive motion search with elastic diamond)[8], 적응적 비대칭 다이아몬드 탐색(AADS: adaptive asymetric diamond search)[9,10], 크로스-다이아몬드 탐색(CDS: cross-diamond search)[11], 작은 크로스-다이아몬드 탐색(SCDS: small-cross-diamond search)[12] 등의 다양한 고속 블록 정합 알고리즘(FBMA: fast block matching algorithm)이 개발되었다. 이들 고속 블록 정합 알고리즘은 주로 탐색영역 내에서 탐색점 후보의 개수를 감소시켜 전체 계산량의 감소를 유도하는 탐색패턴을 사용한다. 탐색패턴이란 블록 정합을 위해 각 탐색단계에서 정합척도를 검사하는 탐색점들을 의미하며, 이 탐색점들 중에서 최소 BDM (minimum block distortion measure) 값을 가지는 위치를 중심으로 다음 단계의 움직임 벡터의 탐색이 수행된다. 따라서 고속 블록 정합 알고리즘에서 사용되는 탐색패턴은 그 모양과 크기에 따라 탐색속도와 화질을 좌우하는 중요한 요소가 될 수 있다.

본 논문에서는 대부분의 움직임 벡터가 탐색영역의 중심 주위에 분포하므로 움직임 벡터를 찾는데 요구되는 계산량을 줄일 수 있도록 초기에 단위 다이아몬드패턴을 이용하여 적은 탐색점으로 움직임이 적은 벡터를 탐색하고 이때 찾지 못한 움직임이 큰 벡터에 대해서는 납작한 육각패턴을 이용하여 고속으로 탐색하는 알고리즘을 제안한다.

2. 기존의 움직임 예측 알고리즘

동영상 내에는 실제로 공간적인 중복성보다는 시간적인 중복성이 훨씬 크기 때문에 시간적인 중복성을 어떻게 찾아내는지의 여부가 동영상 압축의 효율을 높이는 척도가 된다. 시간적인 중복성에 의한 압축은 움직임 벡터를 찾고 그 위치에 있는 이전 블록과의 차이값을 압축/저장하는 방법이다.

움직임 예측은 영상의 블록 또는 화소 단위로 적용되며, 계산 복잡도 및 하드웨어 구현에 있어서 용이한 블록 단위의 움직임 예측이 널리 사용된다. 블록 단위의 움직임 예측은 동일한 블록 내의 화소들은 동일한 움직임을 갖는다는 것과 블록의 탐색을 수평, 수직방향으로만 한정한다는 두 가지 전제조건을 가지므로 영상의 한 프레임을 동일한 크기의 블록들로 나누고 이들의 각 블록들에 대하여 참조 프레임(reference frame)의 탐색영역 내에서 정합오차가 가장 작은 블록이 움직임 벡터로 결정된다. 이러한 영상간의 가장 유사한 블록을 찾는 작업을 움직임 예측이라고 한다.

움직임 예측을 위한 탐색 패턴의 설계를 위해 그림 1과 같이 실험에 적용한 6개의 동영상들에 존재하는 움직임 벡터에 대한 평균적인 분포도를 차트로 나타내었는데 대부분의 움직임 벡터가 탐색 영역의 중심에 분포한다는 것을 볼 수 있다.

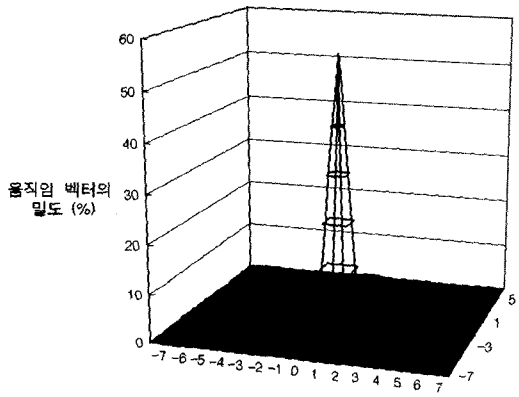
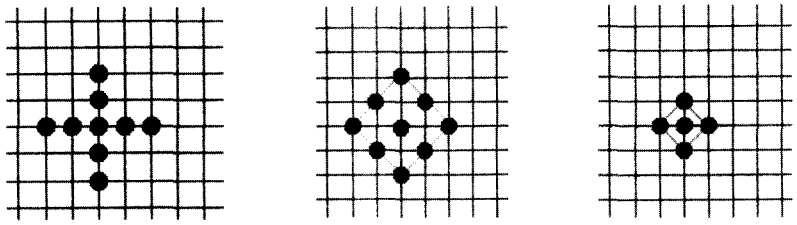


그림 1 전체 실험 동영상의 움직임 벡터 분포도

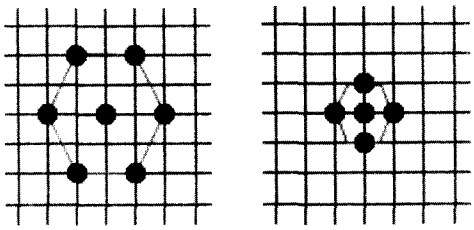
크로스-다이아몬드 탐색(CDS)은 그림 1에서 볼 수 있듯이 움직임 벡터의 대부분이 탐색영역의 중심에 분포하는 특성을 고려하여 그림 2의 (a)와 같이 중심점을 포함하는 초기 크로스패턴을 이용한 탐색을 수행한 후 중심점이 최소 BDM이 아닐 경우 비교적 탐색효율이 높은 것으로 평가된 그림 2의 (b)와 (c)의 다이아몬드패턴을 이용한 탐색을 수행하는 알고리즘이다[11].

크로스-다이아몬드 탐색(CDS) 알고리즘은 그림 2의 (a)와 같이 초기 크로스패턴의 9개의 탐색점을 계산하여 중심점이 최소 BDM이 되면 중심점을 움직임 벡터로 결정하고 탐색을 중단한다(첫 번째 종료조건). 그렇지 않으면 크로스패턴의 중심점으로부터 각각 (-1, -1), (1, -1), (1, 1), (-1, 1)에 위치하는 4개의 후보 탐색점 중 크로스패턴 탐색에서 구해진 최소 BDM과 가까운 두 개의 탐색점을 추가하여 작은 다이아몬드패턴 탐색을 수행한다. 이때 중심점이 최소 BDM이 되면 작은 다이아몬드패턴의 중심점을 움직임 벡터로 결정하고 탐색을 중단한다(두 번째 종료조건). 두 가지의 종료조건을 만



(a) 크로스패턴 (b) 큰 다이아몬드패턴(LDSP) (c) 작은 다이아몬드패턴(SDSP)

그림 2 크로스패턴과 다이아몬드패턴을 결합한 크로스-다이아몬드패턴



(a) 큰 육각패턴(LHEXBSP) (b) 작은 육각패턴(SHEXBSP)

그림 3 육각패턴 탐색 알고리즘의 탐색패턴

족하지 못할 때에는 그림 2의 (b)와 같이 큰 다이아몬드패턴(LDSP: large diamond search pattern)을 적용하여 탐색패턴의 중심점이 최소 BDM이 될 때까지 큰 다이아몬드 형태의 주위 9개의 탐색점을 계산한다. 계산된 최소 BDM 점이 중심점일 때 작은 다이아몬드패턴(SDSP: small diamond search pattern)을 구성하고 최종적으로 BDM 계산을 통해 정합 블록의 움직임 벡터를 구한다. 크로스-다이아몬드패턴을 이용한 탐색 경로는 그림 4와 같다.[11]

육각패턴 탐색(HEXBS)은 그림 3의 (a)와 같이 중심점과 수평방향의 거리가 2인 두 점과 중심점으로부터 거리가 $\sqrt{5}$인 4개의 점으로 이루어진 큰 육각패턴(LHEXBSP: large hexagon-based search pattern)과 그림 3의 (b)와 같은 중심점으로부터 거리가 1인 4개의 탐색점을 갖는 작은 육각패턴(SHEXBSP: small hexagon-based search pattern)을 이용함으로써 다이아몬드 탐색보다 현저한 속도증가를 나타내는 것으로 평가된 탐색 알고리즘이다[6,7].

육각패턴 탐색(HEXBS) 알고리즘은 탐색 영역 내에 미리 정의된 탐색 블록의 중심점(0,0)을 중심점으로 하는 7개의 탐색점을 가지는 큰 육각 패턴을 구성하여 최소 BDM 점을 계산한다. 최소 BDM 점이 큰 육각패턴의 중심점이 될 때까지 이전 단계에서 구해진 최소 BDM 점을 중심으로 하는 큰 육각패턴을 구성하여 탐색을 반복한다. 큰 육각패턴의 중심점이 최소 BDM 점

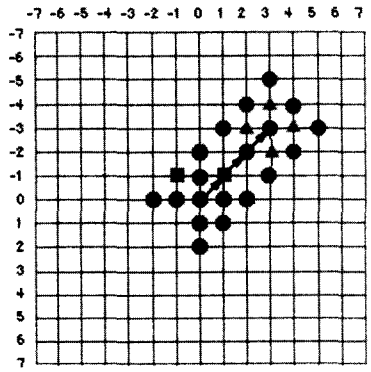


그림 4 크로스-다이아몬드패턴의 탐색경로

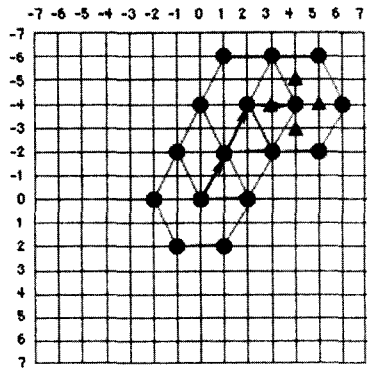


그림 5 육각패턴의 탐색경로

일 때, 중심점 이웃의 4개의 점을 포함하는 작은 육각패턴을 구성하여 최소 BDM 점을 구하고 이 점을 움직임 벡터로 결정한다. 육각패턴을 이용한 탐색 경로는 그림 5와 같다[6,7].

3. 제안한 단위 다이아몬드와 납작한 육각패턴 탐색 알고리즘

동영상에서 인접 프레임간의 시간 간격은 매우 짧기 때문에 단위 프레임의 시간당 움직임 크기 변화량은 일

반적으로 적은 범위로 제한된다고 볼 수 있다. 즉, 연속하는 두 프레임간의 움직임에 많은 시간적 중복성을 가지고 있으므로 참조 프레임의 움직임 정보를 현재 프레임의 동일한 위치 매크로 블록의 탐색 시작점으로 사용함으로써 적은 탐색점들을 사용하여 움직임 벡터를 구할 수 있고, 양호한 보상결과를 얻을 수 있다.

본 논문에서는 참조 프레임의 움직임 정보를 이용하여 해당 프레임에서 해당 블록의 움직임 예측을 위한 초기 탐색점을 설정하고, 움직임 벡터의 분포특성에 맞는 적응적인 탐색패턴을 사용한 새로운 움직임 탐색 알고리즘을 제안한다.

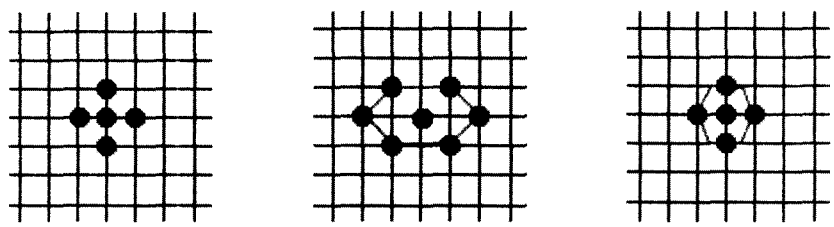
그림 1의 움직임 벡터 분포도에서 살펴본 것과 같이 실험 동영상들에 대해 탐색영역의 거리를 ±7로 두었을 때, 대부분의 움직임 벡터가 탐색영역의 중심 주위에 분포하는 것을 알 수 있었고, 표 1과 같이 움직임 벡터의 분포를 백분율로 나타내 보았을 때 탐색영역의 중심점으로부터 반경 2픽셀 내에 많은 분포를 가지지만 수평방향의 움직임 벡터의 분포가 수직방향의 분포보다 많다는 것을 알 수 있다. 그러므로 기존의 CDS에서 사용한 것과 같은 크로스패턴[11]을 사용한다면 초기 탐색점 수의 증가를 초래하게 되고 이는 전체적인 알고리즘의 속도 저하를 가져올 수 있다. 또한 기존의 HEXBS는 큰 육각패턴[6,7]내의 중심점이 최소 BDM인 경우에 작은 육각패턴[6,7]을 구성하면서 탐색점이 추가될 때, 탐색영역의 일부가 탐색대상에서 제외되므로 정합의 정확도가 저하되고 부정확한 정합으로 인해 잡음이 많이 포함된 움직임 벡터를 추출할 확률이 높아질 수 있다.

본 논문에서는 이러한 기존의 탐색 알고리즘이 가진 문제점들을 해결하기 위해 중심점과 중심점으로부터 반경 1픽셀 내의 이웃점으로 구성된 그림 6의 (a)와 같은 단위 다이아몬드패턴과 그림 6의 (b)와 (c)의 납작한 육각패턴을 결합한 새로운 알고리즘을 제안한다. 제안하는 단위 다이아몬드패턴의 적용에 의하여 탐색영역의 중심 주위에 분포하는 움직임 벡터의 탐색을 적은 탐색점으로 찾을 수 있으며, 단위 다이아몬드패턴에서 탐색되지 못한 움직임 벡터에 대해서는 납작한 육각패턴이 적용되어 빠른 탐색과 화질 향상에 기여하게 된다.

제안하는 단위 다이아몬드와 납작한 육각패턴 탐색은 실험대상 동영상의 움직임 벡터들이 표 1과 같이 탐색영역의 중심점과 이웃하는 4점에 분포할 확률이 평균

표 1 움직임 벡터의 분포(%)

												5	6	7	
0.030	0.016	0.009	0.012	0.010	0.013	0.014	0.070	0.014	0.011	0.011	0.009	0.007	0.014	0.026	
0.015	0.011	0.008	0.008	0.006	0.011	0.012	0.053	0.014	0.008	0.010	0.003	0.007	0.006	0.014	
0.019	0.012	0.011	0.015	0.009	0.014	0.016	0.076	0.014	0.014	0.014	0.011	0.009	0.007	0.017	
0.032	0.013	0.020	0.015	0.021	0.023	0.027	0.105	0.023	0.019	0.018	0.013	0.010	0.007	0.018	
0.050	0.024	0.027	0.022	0.027	0.028	0.055	0.149	0.036	0.036	0.024	0.017	0.014	0.016	0.037	
0.061	0.028	0.027	0.022	0.052	0.091	0.187	0.554	0.107	0.083	0.054	0.040	0.022	0.019	0.061	
0.125	0.045	0.050	0.062	0.112	0.275	1.364		1.031	0.307	0.169	0.136	0.101	0.084	0.196	
0.731	0.258	0.408	0.312	0.642	1.227				6.274	1.162	1.519	1.275	0.247	0.779	
0.227	0.077	0.087	0.104	0.138	0.239	0.639		0.669	0.205	0.109	0.057	0.045	0.036	0.142	
0.107	0.049	0.048	0.060	0.056	0.075	0.123	0.471	0.086	0.071	0.051	0.029	0.029	0.030	0.071	
0.022	0.010	0.013	0.017	0.029	0.064	0.138	0.202	0.043	0.026	0.038	0.017	0.020	0.019	0.058	
0.012	0.009	0.010	0.013	0.012	0.023	0.089	0.179	0.018	0.017	0.017	0.023	0.011	0.010	0.034	
0.016	0.008	0.011	0.013	0.011	0.014	0.031	0.091	0.019	0.014	0.008	0.010	0.009	0.009	0.026	
0.017	0.008	0.010	0.011	0.011	0.020	0.240	0.074	0.040	0.007	0.005	0.008	0.006	0.008	0.026	
0.039	0.010	0.011	0.009	0.018	0.018	0.055	0.373	0.040	0.047	0.016	0.010	0.014	0.014	0.071	



(a) 단위 다이아몬드패턴 (b) 큰 납작한 육각패턴 (c) 작은 육각패턴

그림 6 단위 다이아몬드패턴과 수평 방향(X축)의 움직임 벡터를 고려한 납작한 육각패턴

약 71.22%라는 사실을 이용하여 초기 탐색점들을 배치하였고, 탐색영역의 중심에서 수평방향으로 존재하는 움직임 벡터를 빨리 탐색하여 점진적으로 움직임 예측의 확률을 높일 수 있도록 추가 탐색점들을 적절히 배치하였다.

본 논문에서 제안한 탐색 패턴을 이용하여 움직임 벡터를 탐색하는 절차를 표 2와 같이 의사코드로 나타내었다.

제안하는 알고리즘은 표 2의 의사코드에서 나타낸 바와 같이 우선 탐색 영역의 (0,0)을 중심점으로 그림 7의 (a)와 같이 1픽셀 간격의 4개의 추가점을 포함하는 5개의 탐색점으로 구성된 단위 다이아몬드패턴의 탐색점에 대하여 최소 절대값 오차의 합(SAD : sum of absolute difference)을 계산한다. 이 때, 중심점이 최소 SAD 점이면 중심점을 움직임 벡터(MV=(0,0))로 결정하고 탐색을 중단한다. 중심점이 최소 SAD 점이 아니면 계산된 최소 SAD 점을 중심점으로 하는 그림 7의 (b)와 같은 큰 납작한 육각패턴을 구성하여 최소 SAD를 계산한다. 새로 계산된 최소 SAD 점이 중심점에 위치할 때까지 그림 7의 (c)와 같이 큰 납작한 육각 패턴의 구성과 최소 SAD 계산 과정을 반복하며, 최종적으로 최소 SAD 점이 중심점이 되었을 때 그림 7의 (d)와 같이 중심점으로부터 반경 1픽셀에 해당되는 이웃 4 점을 포함하는 작은 육각패턴을 구성하여 최소 SAD를 계산하고, 이 단계에서 구해진 최소 SAD 점이 움직임 벡터가 된다.

표 2 의사코드 : 단위 다이아몬드와 납작한 육각패턴 탐색 알고리즘

```
For each MacroBlock in the Current Frame
    Search Origin <- MB(0,0)
    Make Unit Diamond pattern and Calculate Minimum
    SAD point
    IF Minimum SAD = Search Origin in Unit Diamond
    pattern
        MV <- (0,0), Stop;
    Else
        Search Origin <- Minimum SAD(x,y)
        Make Large Flat Hexagon pattern and Calculate
        Minimum SAD point
        While Minimum SAD = Search Origin in Large
            Flat Hexagon pattern
            Search Origin <- Minimum SAD(x,y)
            Make Large Flat Hexagon pattern and
            Calculate Minimum SAD point
        End While
        Search Origin <- Minimum SAD(x,y)
        Make Small Flat Hexagon pattern and Calculate
        Minimum SAD point
        MV <- Minimum SAD(x,y), Stop;
End For
```

4. 실험 결과

제안된 알고리즘의 성능을 평가하기 위하여 4개의 CIF(352×288) 영상 Akiyo, Foreman, Coastguard, Miss America와 SIF(352×240) 영상 Football, 그리고 QCIF(176×144) 영상 Table에 대해 각각 100프레임씩을 대상으로 실험하였고, 비교 탐색 알고리즘으로는 FS, DS, CDS, SCDS, HEXBS, 그리고 제안한 탐색 알고리즘을 사용하였다. 움직임 예측에 사용된 매크로 블록의 크기는 16×16 픽셀이며, 탐색영역의 변위는 ±7을 적용하여 Pentium Ⅳ 1.6GHz와 256MB 메모리가 장착된 컴퓨터상에서 실험을 수행하였다.

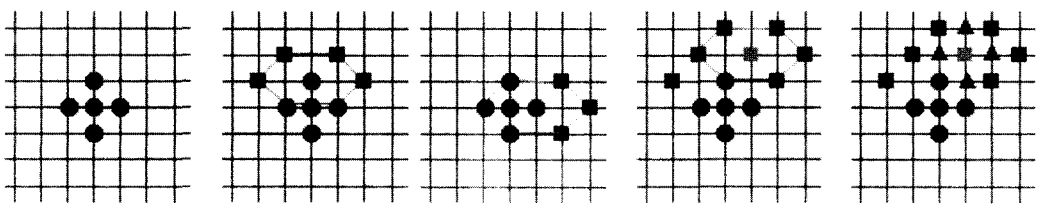
성능 비교 평가 함수로는 영상 화질의 품질을 평가하기 위해 평균 절대값 오차(MAD: mean absolute difference)와 평균 제곱 오차(MSE: mean squared error)를 이용한 PSNR(peak signal-to-noise ratio)를 이용하였으며, 정합 오차 측정 함수로는 절대값 오차의 합(SAD: sum of absolute difference)을 이용하였다. 또한 제안하는 알고리즘의 성능 향상을 측정하기 위해 블록 당 탐색점의 수를 기존 알고리즘들과 비교하였다.

탐색 알고리즘의 성능을 비교평가하기 위해 사용한 함수인 MAD와 MSE를 이용한 PSNR은 각각 식 (1)~(2)와 같다.

$$MAD=\left(\frac{1}{M\times N}\right)\sum_{x=1}^{M}\sum_{y=1}^{N}|O_t(x,y)-E_t(x,y)| \quad (1)$$

$$PSNR = 10\log_{10}\frac{255^2}{MSE}, \quad MSE = \left(\frac{1}{M\times N}\right)\sum_{x=1}^{M}\sum_{y=1}^{N}[O_t(x,y)-E_t(x,y)]^2 \quad (2)$$

식 (1)과 (2)에서 M과 N은 각각 영상의 가로와 세로



(a) 1단계 (b) 2단계 (c) 3단계 (d) 4단계

그림 7 제안하는 알고리즘의 탐색경로

의 크기를 나타내며, $O_t(x, y)$는 원영상의 화면을 나타내고, $E_t(x, y)$는 움직임 예측 화면을 나타낸다.

블록의 정합오차를 측정하기 위한 함수 SAD는 식 (3)과 같다.

$$SAD = \sum_{k=1}^{MB} \sum_{l=1}^{MB} |I_t(k, l) - I_{t-1}(k+i, l+j)| \quad (3)$$

식 (3)에서 MB는 매크로 블록의 가로와 세로의 크기를 나타내며, $I_t(k, l)$은 현재 프레임을 나타내고, $I_{t-1}(k+i, l+j)$은 이전 프레임을 나타낸다.

실험영상에 대한 실험 결과는 표 3, 그림 8, 그리고 그림 9에 각각 나타내었다. 표 3에서는 실험영상별로 전체 프레임에 대한 기존의 탐색 알고리즘과 제안하는 탐색 알고리즘의 실험결과를 나타내었으며 그림 8과 그림 9에서는 움직임이 많이 포함된 영상과 움직임이 적게 포함된 영상들 중 하나씩을 선택해서 두 실험영상의 프레임별 블록 당 평균 탐색점 수와 프레임별 픽셀 당 MAD 값을 각각 그래프로 나타내었다.

표 3에서 Ns 항목은 각 실험영상에서 각 블록의 움직임 벡터 예측시 사용된 평균 탐색점의 수를 나타내었고 이를 이용하여 계산된 속도 향상의 비율을 SpUp 항목으로 나타내었다. 또한 MAD 항목은 원영상과 움직임 예측에 의해 생성된 영상의 각 픽셀 사이의 평균 절대값 오차를 나타내며 PSNR 항목은 원영상과 움직임 예측에 의해 생성된 영상의 각 픽셀 사이의 MSE를 이용해 계산된 PSNR[dB]을 나타내었다.

제안하는 탐색 알고리즘의 탐색점의 수와 탐색 속도의 성능 향상에 대해 실험영상별로 분석해보면 다음과 같다. 실험영상 Akiyo(CIF)에서는 CDS와 HEXBS에 비해 탐색점 수가 각각 42%, 51%가 감소되었고, 탐색속도에 있어서는 각각 72%, 104%의 향상을 보였다. 실험영상 Foreman(CIF)에서는 CDS와 HEXBS에 비해 탐색점 수가 각각 19%, 11%가 감소되었고, 탐색속도에 있어서는 각각 23%, 13%의 향상을 보였다. 실험영상 Coastguard(CIF)에서는 CDS와 HEXBS에 비해 탐색점 수가 각각 22%, 12%가 감소되었고, 탐색속도에 있어서는 각각 29%, 14%의 향상을 보였다. 실험영상 Miss America(CIF)에서는 CDS와 HEXBS에 비해 탐색점 수가 각각 13%, 17%가 감소되었고, 탐색속도에 있어서는 각각 15%, 20%의 향상을 보였다. 실험영상 Football(SIF)에서는 CDS와 HEXBS에 비해 탐색점 수가 각각 23%, 17%가 감소되었고, 탐색속도에 있어서는 각각 31%, 21%의 향상을 보였다. 실험영상 Table (QCIF)에서는 CDS와 HEXBS에 비해 탐색점 수가 각각 23%, 20%가 감소되었고, 탐색속도에 있어서는 각각 29%, 24%의 향상을 보였다.

전체적인 실험결과를 볼 때, 모든 실험영상에 대해서 제안하는 탐색 알고리즘이 기존의 탐색 알고리즘들에 비해 탐색점의 수가 현저하게 감소함으로써 탐색속도의 향상을 가져오는 것을 알 수 있다. 전체적으로 제안하는 알고리즘이 CDS에 비해 탐색점의 수가 약 13~42% 정

표 3 각 실험영상에 대한 실험결과

(a) 블록 당 탐색점 수와 속도향상 비율

BMA	Akiyo (CIF)		Foreman (CIF)		Coastguard (CIF)		Miss America (CIF)		Football (SIF)		Table (QCIF)	
	Ns	SpUp	Ns	SpUp	Ns	SpUp	Ns	SpUp	Ns	SpUp	Ns	SpUp
FS	204.28	1.000	204.28	1.000	204.28	1.000	204.28	1.000	202.05	1.000	184.56	1.000
DS	12.29	16.628	15.99	12.775	16.74	12.201	16.53	12.360	16.17	12.494	13.83	13.344
CDS	8.72	23.422	13.48	15.155	14.67	13.925	11.71	17.447	13.41	15.063	11.63	15.863
SCDS	5.08	40.250	12.52	16.316	14.30	14.286	10.81	18.906	11.66	17.328	10.19	18.110
HEXBS	10.35	19.744	12.36	16.534	12.97	15.751	12.18	16.770	12.38	16.318	11.19	16.490
Proposed	**5.08**	**40.227**	**10.98**	**18.610**	**11.39**	**17.940**	**10.14**	**20.138**	**10.27**	**19.674**	**8.99**	**20.522**

(b) 성능 비교 평가 함수의 결과값

BMA	Akiyo (CIF)		Foreman (CIF)		Coastguard (CIF)		Miss America (CIF)		Football (SIF)		Table (QCIF)	
	MAD	PSNR	MAD	PSNR	MAD	PSNR	MAD	PSNR	MAD	PSNR	MAD	PSNR
FS	0.605	42.173	2.909	32.888	5.444	29.089	1.914	39.055	10.242	22.392	5.021	26.076
DS	0.606	42.150	2.998	32.619	5.605	28.637	2.003	38.712	10.758	21.857	5.160	25.745
CDS	0.606	42.127	3.086	32.215	5.762	28.285	1.944	38.955	11.207	21.494	5.229	25.597
SCDS	0.606	42.127	3.087	32.207	5.781	28.241	1.946	38.950	11.259	21.433	5.259	25.524
HEXBS	0.619	41.849	3.279	31.887	5.741	28.356	2.100	38.222	11.117	21.628	5.506	25.278
Proposed	**0.606**	**42.147**	**3.015**	**32.486**	**5.707**	**28.405**	**1.944**	**38.960**	**11.046**	**21.567**	**5.259**	**25.532**

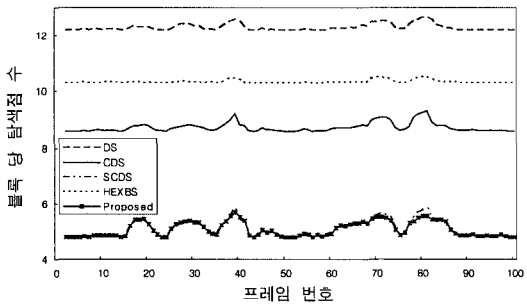
도 감소되었고 탐색속도 면에서는 약 15~72% 정도의 성능 향상을 나타내었다. 또한 육각패턴 탐색에 대해서는 탐색점의 수가 약 11~51% 정도 감소하였고 탐색속도 면에서는 약 13~104% 정도의 성능 향상을 나타내었다. 대부분의 실험영상이 비슷한 성능 향상을 나타낸 가운데 실험영상 Akiyo(CIF)의 경우는 다른 실험영상들에 비해 현격한 성능 향상을 나타내었는데 이는 움직임이 아주 적게 포함된 경우이므로 제안하는 알고리즘의 탐색과정 중 단위다이아몬드패턴 탐색에서 움직임 벡터가 대부분 결정되어짐으로서 더 많은 성능 향상을 가지는 것으로 분석된다.

영상의 화질에 대한 성능 비교 평가 함수로 사용된 MAD와 MSE를 이용해 계산된 PSNR의 실험결과에 대해 각 실험영상별로 분석해 보면 다음과 같다. 실험영상 Akiyo(CIF)에서 CDS에 대해서는 MAD는 같고 PSNR이 0.02dB의 향상을 보였고 HEXBS에 비해서는 MAD가 0.013이 감소되었고, PSNR은 0.298dB의 향상을 보였다. 실험영상 Foreman(CIF)에서는 CDS와 HEXBS에 비해 MAD가 각각 0.071, 0.264 정도 감소하였고, PSNR은 각각 0.271dB, 0.599dB의 향상을 보였다. 실험영상 Coastguard(CIF)에서는 CDS와 HEXBS에 비해 MAD가 각각 0.055, 0.034가 감소되었고, PSNR은 0.12dB, 0.049dB 정도의 향상을 보였다. 실험영상 Miss America(CIF)에서는 CDS에 대해서는 MAD는 같고 HEXBS에 비해서는 MAD가 0.156 정도 감소되었고, PSNR은 0.005dB, 0.738dB의 향상을 보였다. 움직임이 많이 포함된 실험영상 Football(SIF)에서는 CDS와 HEXBS에 비해 MAD가 각각 0.161, 0.071이 감소되었고, PSNR은 CDS에 대해서는 0.073dB의 향상을 보였으나 HEXBS에 대해서는 오히려 0.061dB의 감소를 보였다. 이 결과는 HEXBS가 움직임이 많이 포함된 영상의 움직임 예측에 뛰어난 알고리즘이란 것을 증명해주는 결과이다. 또한 수직적인 움직임을 많이 포함하고 있는 실험영상 Table(QCIF)에서는 MAD가 CDS에 대해서는 0.03 증가하였고 HEXBS에 대해서는 0.247 정도 감소되었다. PSNR은 CDS에 대해서는 오히려 0.065가 증가되었고 HEXBS에 대해서는 0.254dB의 향상을 보였다. 이는 수평적인 움직임 분포를 많이 고려한 탐색패턴을 사용함으로 인해 발생되는 근소한 차이라고 분석된다.

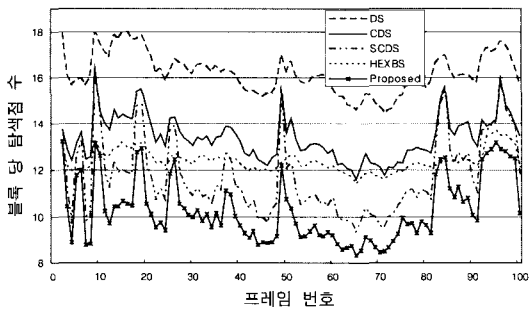
전체적인 실험결과를 보면 제안된 탐색 알고리즘보다 18~40배 이상 많은 탐색점을 사용하여 움직임 벡터를 예측하는 전역 탐색과도 많은 차이가 나지 않는 것을 볼 수 있다. 제안된 탐색 알고리즘은 비교대상이 된 HEXBS나 CDS에 비해 우수하거나 근소한 차이를 가지는 움직임 예측성능을 가지면서, 탐색속도 면에서는 현저히 우수한 성능 향상을 보였다.

그림 8에서는 실험에 사용된 움직임이 적게 포함된 Akiyo(CIF) 영상과 움직임이 많이 포함된 Football (SIF) 영상에 대한 프레임별 블록 당 평균 탐색점 수에 대한 그래프를 볼 수 있다. 움직임이 적게 포함된 영상의 경우에는 프레임간의 시간적 상관성이 많이 존재하므로 기존의 탐색 알고리즘 특히 직접적인 비교대상인 CDS와 HEXBS에 비해 움직임 예측에 필요한 탐색점의 수가 현저하게 감소된 것을 볼 수 있다. 움직임이 많이 포함된 영상의 경우에는 영상의 프레임 간에 존재하는 시간적 상관성이 상대적으로 적어서 시간적 상관성이 많은 영상들보다는 탐색점의 수가 증가하게 된다. 하지만 제안하는 탐색 알고리즘은 시간적 상관성이 적은 영상에서도 CDS에 대해서 약 22~23%, HEXBS에 대해서는 12~20% 정도의 탐색점 수의 감소를 나타내었다. 제안하는 탐색 알고리즘이 기존의 탐색 알고리즘에 비해 움직임 예측에 필요한 탐색점 수를 현저히 감소시킴으로써 탐색속도 측면에서 높은 성능 향상을 보인다는 것을 나타낸 것이다.

그림 9의 결과는 실험에 사용된 움직임이 적게 포함된 Akiyo(CIF) 영상과 움직임이 많이 포함된 Football(SIF) 영상에 대한 프레임별 픽셀 당 평균 절대값 오차를 나타낸 그래프로서 전역 탐색과 차이가 작은 탐

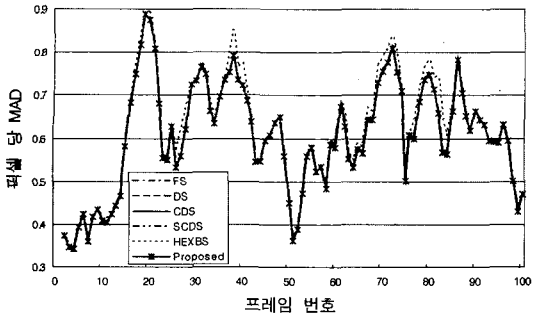


(a) 움직임이 적게 포함된 Akiyo(CIF) 영상

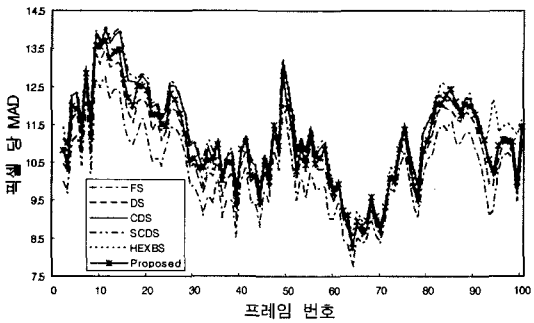


(b) 움직임이 많이 포함된 Football(SIF)영상

그림 8 실험영상의 프레임별 블록 당 평균 탐색점 수



(a) 움직임이 적게 포함된 Akiyo(CIF) 영상



(b) 움직임이 많이 포함된 Football(SIF)영상

그림 9 실험영상의 프레임별 픽셀 당 MAD

색 알고리즘이 움직임 벡터 예측성능이 우수함을 나타내는 것이다. 그림 8의 실험결과와 마찬가지로 움직임이 많이 포함된 영상과 움직임이 적게 포함된 영상들이 조금씩 다른 결과를 나타내지만 결과적으로는 제안하는 탐색 알고리즘이 CDS를 적용한 결과와 근소한 차이만을 가지며 HEXBS보다는 우수한 실험결과를 나타냈다.

5. 결 론

본 논문에서는 동영상 프레임의 시간적 상관성에 따라 움직임 벡터의 분포를 바탕으로 하여 적응적으로 탐색원점과 탐색패턴을 바꾼 새로운 탐색 알고리즘을 제안하였다. 초기에 단위 다이아몬드패턴을 이용한 빠른 탐색과 수평방향으로 많이 분포하는 움직임 벡터의 빠른 탐색을 결합함으로써 실험을 통하여 알 수 있듯이 제안된 알고리즘은 움직임 예측에 필요한 탐색점의 수를 약 11~51% 감소시킴으로써 탐색속도 면에서 높은 향상을 보였고, 화질 면에서도 가장 우수한 성능을 가지는 전역 탐색에 근접하는 결과를 얻을 수 있었다. 육각 패턴 탐색과 비교하였을 경우에 움직임 보상 예측된 화질에 있어서 약 0.05~0.74dB 정도의 성능을 향상시켰으며 다른 기존의 탐색 알고리즘들과 비교할 때에도 우수하거나 근사한 성능을 보였다.

움직임 벡터의 예측 과정에서 발생할 수 있는 예외적인 요소를 모두 고려하여 가장 적응적인 현재 프레임의 매크로 블록 탐색원점을 예측하고 본 논문에서 제안한 탐색 알고리즘을 사용하여 움직임 예측을 한다면 보다 빠르게 움직임 벡터를 찾을 수 있을 것이며, 우수한 보상 결과를 얻을 수 있을 것으로 기대된다. 또한 일정한 시간간격을 두고 연속적으로 추출되는 움직임 벡터의 크기를 일정하게 조절할 수 있는 시간적인 동기화에 대한 연구도 보강될 필요가 있다.

참 고 문 헌

[1] T. Koga, K. Iinuma, A. Hirano, Y. Iijima, and T. Ishiguro, "Motion-compensated Interframe Coding for Video Conferencing," in Proc. National Telecommunications Conference, New Orleans, LA, pp.G5.3.1-G5.3.5, Nov. 1981.

[2] R. Li, Bing Zeng, "A New Three-Step Search Algorithm for Fast Block-Matching Motion Estimation," IEEE Transactions on Circuits & System for Video Technology, Vol. 4, No. 4, pp.438~442, Aug., 1994.

[3] L. M. Po, W.C. Ma, "A Novel Four-Step Search Algorithm for Fast Block-Matching Motion Estimation," IEEE Transactions on Circuits & System for Video Technology, Vol. 6, No. 3, pp.313~317, June 1996.

[4] 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.

[5] Sung-Chul Shin, Hyunki Baik, Myong-Soon Park, Dong Sam Ha, "A center-biased hybrid search method using plus search pattern for block motion estimation," Circuits and Systems, 2000. Proceedings. ISCAS 2000 Geneva. The 2000 IEEE International Symposium on , Vol. 4, pp.309~312, May, 2000.

[6] Ce Zhu, Xiao Lin, Lap-Pui Chau, Keng-Pang Lim, Hock-Ann Ang, Choo-Yin Ong, "A novel hexagon-based search algorithm for fast block motion estimation," Acoustics, Speech, and Signal Processing, 2001. Proceedings. (ICASSP '01). 2001 IEEE International Conference on, Vol.3, 1593~1596, May 2001.

[7] Ce Zhu, Xiao Lin, Lap-Pui Chau, "Hexagon-Based Search Pattern for Fast Block Motion Estimation," IEEE Transactions on Circuits & System for Video Technology, Vol. 12, No. 5, pp.349~355, May 2002.

[8] Weiguo Zheng, I. Ahmad, Ming Lei Liou, "Adaptive motion search with elastic diamond for MPEG-4 video coding," Image Processing, 2001. Proceedings. 2001 International Conference on,

Vol.1, 377~380 Oct. 2001.
[9] T. Sappasitwong, S. Aramvith, S. Jitapunkul, A. Tamtrakarn, P. Kitti-punyangam, H. Kortrakulkij, "Adaptive asymmetric diamond search algorithm for block-based motion estimation," Video/Image Processing and Multimedia Communications 4th EURASIP-IEEE Region 8 International Symposium on VIPromCom, pp.16~19 Jun., 2002.
[10] T. Sappasitwong, S. Aramvith, S. Jitapunkul, A. Tamtrakarn, P. Kitti-punyangam, H. Kortrakulkij, "Adaptive asymmetric diamond search algorithm for block-based motion estimation," Digital Signal Processing, 2002. DSP 2002. 2002 14th International Conference on, Vol. 2, pp.563~566, Jul., 2002.
[11] C. H. Cheung, L. M. Po, "A Novel Cross-Diamond Search Algorithm for Fast Block-Matching Motion Estimation," IEEE Transactions on Circuits & System for Video Technology, Vol. 12, No. 12, pp.1168~1177, Dec., 2002.
[12] C. H. Cheung, L. M. Po, "A Novel Small-Cross-Diamond Search Algorithm for Fast Video Coding and Videoconferencing Applications," Image Processing. 2002. Proceedings. 2002 International Conference on, Vol. 1, pp.I-681-I-684, Sep., 2002.

남 현 우

1993년 아주대학교 공과대학 전자계산학과(학사). 1996년 아주대학교 대학원 컴퓨터공학과(석사). 1996년~현재 아주대학교 대학원 컴퓨터공학과 박사과정. 2003년~현재 (주)이테크 기술이사. 관심분야는 컴퓨터그래픽스, 이미지프로세싱 및 멀티미디어

위 영 철

1980년 연세대학교 수학과(학사). 1982년 SUNY at Albany Computer Science Department(학사). 1984년 SUNY at Albany Computer Science Department (석사). 1989년 SUNY at Albany Computer Science Department(박사). 1990년~1995년 삼성종합기술원 수석연구원. 1995년~1998년 현대전자 부장. 1998년~현재 아주대학교 정보통신대학 조교수. 관심분야는 컴퓨터그래픽스, 이미지프로세싱 및 알고리즘

김 하 진

1962년 서울대학교 문리과대학 수학과(학사). 1978년 프랑스 Grenoble 1대학교 대학원 응용수학과(석사). 1980년 프랑스 Saint-Etienne 대학교 대학원 응용수학과(박사). 1991년~1992년 한국정보과학회 회장. 1992년~현재 국제정보올림피아드 추진위원장. 1992년~현재 JTC1/IEC SC24(그래픽스 표준화) 국내위원회 위원장. 1974년~현재 아주대학교 정보통신대학 교수. 관심분야는 컴퓨터그래픽스, 이미지프로세싱 및 응용수학