TDS 기법을 이용한 움직임 벡터 추정

김 미 영[†]·정 미 경[††]

요 약

본 논문은 움직임 벡터를 보다 빠르고 정확하게 추정해나가는 탐색 방법으로 상·좌·우 3 방향을 고려한 TDS(T-Shape Diamond Search) 알고리즘을 제안하였다. 이 방법에서는 실제 움직임 벡터가 탐색 영역의 중심과 상·하·좌·우 방향에 집중되어 있는 특성을 이용하여 먼저 탐색 원점을 중심으로 상·하·좌·우 4 방향으로 탐색 점을 배치한 후 블록 정합을 실행한다. 이들 중 정합 오차가 가장 작은 지점을 기준점으로 상 방향으로 탐색 점을 확장하여 정합 오차를 측정해보고 기준점보다 오차가 작으면 계속 상 방향으로 확장해 나가고 그렇지 않으면 기준점을 중심으로 좌우 두 점 중 정합오차가 작은 점을 선택한다. 예측된 방향으로 위의 과정을 반복하며 움직임을 추정한다. 특히 움직임이 십자방향에 집중되는 영상의 경우 접근이 빠르고, 단계적으로 움직임 가능성이 낮은 부분을 탐색 대상에서 제외해 나감으로써 탐색이 비교적 빠르고 정확하게 이루어진다. 이 방법은 기존의 부분 최적 탐색 기법인 NTSS, DS, 그리고 HEXBS 등의 탐색법과 비교할 때 유사한 화질을 유지하면서도 탐색 점수에서는 평균 38%의 개선된 결과를 얻을 수 있었다. 특히 움직임이 적은 영상에서의 탐색 점수는 50%의 향상된 결과를 얻었다.

Motion Vector Estimation using T-shape Diamond Search Algorithm

Miyoung Kim[†] · Migyoung Jung[††]

ABSTRACT

In this paper, we proposed the TDS(T-shape Diamond Search) based on the directions of above, below, left and right points to estimate the motion vector fast and more correctly. In this method, we exploit the facts that most motion vectors are enclosed in a circular region with a radius of 2 fixels around search center(0,0). At first, the 4 points in the above, below, left and right around the search center is calculated to decide the point of the MBD(Minimum Block Distortion). And then an above point of the MBD is checked to calculate the SAD. If the SAD of the above point is less than the previous MBD, this process is repeated. Otherwise, the right and left points of MBD are calculated to decide the points that have the MBD between right point and left point. Above processes are repeated to the predicted direction for motion estimation. Especially, if the motions of image are concentrated in the crossing directions, the points of other directions are omitted. As a result, we can estimate motion vectors fast. Experiments show that the speedup improvement of the proposed algorithm over Diamond Search algorithm(DS) and HEXagon Based Search(HEXBS) can be up to 38~50% while maintaining similar image quality.

키워드 : 움직임 추정(Motion Estimation), 움직임 벡터(Motion Vector), 블록 정합 알고리즘(Block Matching Algorithm)

1. 서 론

최근 정보 통신의 발달과 더불어 동영상 정보의 고속 전송에 대한 요구가 증가하고 있다. 따라서 동영상 정보의 고속 전송을 하기 위해서는 동영상 내에 존재하는 중복된 데이터를 제거하여 전송 데이터양의 감소시킬 수 있는 데이터 압축이 필요하다. MPEG-1/2/4, H.261, 그리고 H.263과 같은 영상 압축 표준안들은 블록 정합 알고리즘(Block Matching Algorithm : BMA)을 기반으로 하는 움직임 추정 기법을 사용하여 동영상 내에 존재하는 중복된 데이터를 제거함으로써 데이터를 압축한다.

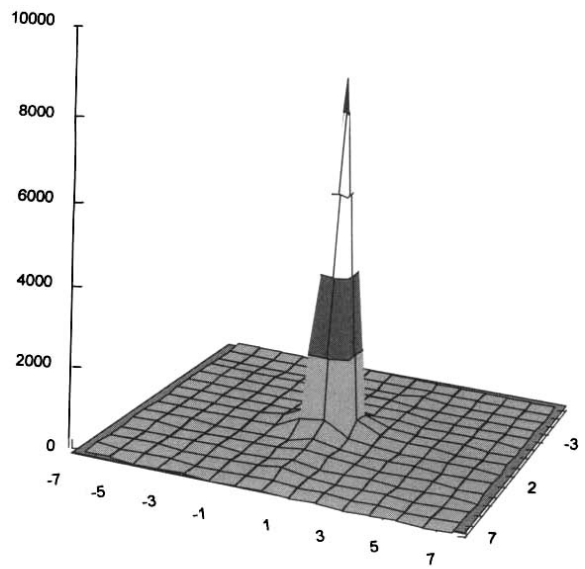
블록 정합 기법은 영상의 한 프레임을 동일한 블록으로 나누고, 이들의 각 블록에 대하여 참조 프레임의 탐색 영역 내에서 정합 오차가 가장 작은 블록을 찾는 기법으로, 현재 프레임의 한 블록과 참조 프레임 내에서 가장 정합이 잘되는 블록간의 위치 차이를 움직임 벡터라 한다. 가장 간단한 블록 정합 알고리즘으로는 전역 탐색(Full Search) 기법이 있는데 이 기법은 움직임 벡터를 추정하기 위해서 기준 블록과 탐색 영역내의 모든 화소와 정합하여 최소 정합 오차를 가진 점의 변위를 움직임 벡터로 추정한다. 전역 탐색

† 정 회 원 : 전남도립남도대학 컴퓨터정보통신과 교수
†† 정 회 원 : 전남대학교 대학원 전산통계학과, 교신저자
논문접수 : 2003년 9월 15일, 심사완료 : 2004년 4월 27일

기법은 탐색 범위 내에서 가장 적합한 움직임 벡터를 구할 수 있지만 계산양이 많으므로 실시간 비디오 코딩 응용 분야 및 소프트웨어 구현에 많은 어려움을 가지고 있다.

이러한 문제점을 해결하기 위하여 여러 가지 고속 블록 정합 기법(Fast Block Matching Algorithm : FBMA)들이 개발되었는데, 대표적인 고속 블록 정합 기법에는 3단계 탐색(Three Step Search : TSS)[1], 새로운 3단계 탐색(New Three Step Search : NTSS)[2], 2차원 로그형 탐색(2 Dimension LOGarithmic search : 2DLOG)[3], 4단계 탐색(Four Step Search : FSS)[4], 다이아몬드 탐색(Diamond Search : DS)[5], 2단계 탐색(2 Step Search : 2SS)[6], 그리고 육각형 탐색(Hexagon-based Search : HEXBS)[7] 등이 있다.

위의 기법들[1-7]은 전역 탐색 기법보다는 수행시간이 짧고 그것에 근접하는 성능을 얻기 위한 것들이다. 또한 위의 기법들은 (그림 1)과 같이 영상의 대부분의 움직임 방향이 중심에 분포하기 때문에, 탐색 영역의 중심에 실제 움직임 벡터가 집중되어 있다는 성질을 이용하고 있다. 그러나 위의 기법들은 움직임 벡터를 찾는데 많은 탐색 점들을 사용한다. 특히 다이아몬드 탐색 기법은 움직임 벡터의 분포가 다이아몬드라는 것과 움직임 벡터가 탐색 영역의 중심에 치우쳐 존재한다는 사실을 이용한 기법으로 움직임이 작거나 없는 영상에서는 좋은 화질을 유지하는데, 움직임이 많은 영상에서는 현재 프레임의 해당 블록에 대한 움직임 정보를 가지고 있지 않기 때문에 움직임 벡터를 찾는데 많은 탐색 점을 사용하고 추정된 움직임 벡터도 정확하지 못한 문제점들을 가지고 있다.



(그림 1) Table 영상의 움직임 벡터의 분포도

이러한 문제점을 해결하기 위하여 새로운 탐색 패턴을 제안한다. 이 탐색 패턴은 (그림 1)과 같이 영상의 움직임이 중심으로부터 상·하·좌·우 방향으로 일어나는 특성을 이용하여 탐색 영역에서 탐색 원점을 중심으로 8개의 탐색 점들을 배치하는 다이아몬드 기법과 다르게 상·하·좌·우 방향으로 4개의 탐색 점들을 배치는 한 후, 정합 오차가 가장 작은 지점을 기준으로 상·좌·우 3 방향으로 탐색 점을 확장하여 블록 정합 오차를 측정해 보고 세 방향 중 오차가 가장 작은 한 방향을 움직임 방향으로 예측한다. 예측된 방향으로 위의 과정을 반복하며 움직임을 추정함으로써, 움직임 가능성이 낮은 부분을 탐색 대상에서 제외시켜서 탐색이 빠르고 정확하게 한다. 기존의 탐색 방법인 NTSS, DS, 그리고 HEXBS 등의 탐색법과 비교할 때 유사한 화질을 보이면서도 탐색 포인트 수가 줄어 움직임 벡터를 보다 빠르고 정확하게 추정해 나가는 탐색패턴으로, 다이몬드 탐색 기법을 수정한 TDS 알고리즘을 제안한다. 본 논문의 구성은 다음과 같다. 2장에서는 움직임 추정 기법에 대하여 설명하고, 3장에서는 제안된 기법을 기술하였다. 그리고 4장에서는 기존 탐색 기법들과 성능을 비교한 후, 마지막으로 5장에서 결론을 맺는다.

2. 움직임 추정(Motion Estimation)기법

움직임 추정 방법은 추정의 기본 단위에 따라 크게 화소 순환 알고리즘(Pel Recursive Algorithm : PRA)과 블록 정합 알고리즘(Block Matching Algorithm : BMA)으로 나누어진다. 현재 많은 비디오 코딩에서, 블록 정합 알고리즘은 화소 순환 알고리즘에 비하여 데이터의 흐름의 규칙성, 계산의 복잡도, 하드웨어의 구현이 용이하기 때문에 움직임 벡터 추정 및 움직임 보상 부호화에서 많이 사용되고 있다.

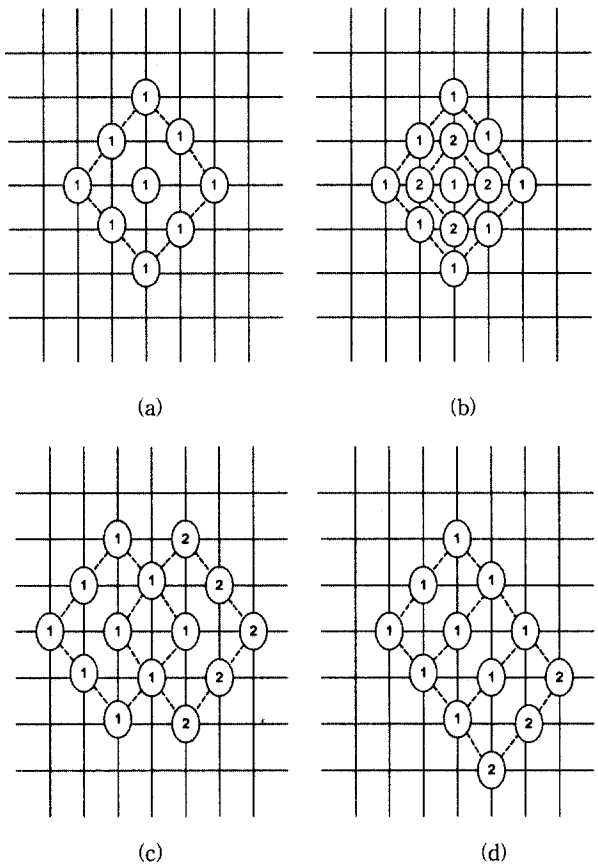
블록 단위의 움직임 추정은 두 가지 전제 조건을 가지고 있는데, 동일한 블록내의 화소들은 동일한 움직임을 갖는다는 것과 블록들은 수평, 수직으로만 움직인다는 것이다. 즉 블록 정합 기법은 영상의 한 프레임을 동일한 크기의 블록들로 나누고 각각의 블록에 대하여 참조 프레임(Reference Frame)의 탐색 영역 내에서 정합 오차가 가장 작은 블록을 찾는다. 이때 현재 프레임(Current Frame)의 해당 블록과 참조 프레임 내에서 블록 정합 오차가 가장 작은 블록과의 블록간의 위치 차이를 움직임 벡터라 한다.

움직임 벡터를 추정하기 위하여 많은 고속 블록 정합 기법들이 제안되었는데, 3단계 탐색 기법은 넓은 영역에 걸쳐

몇 개의 탐색 점을 조사한 후 점차 탐색 변위를 좁혀 나가는 방식으로 3단계 만에 움직임 벡터를 추정하는 기법이고, 새로운 3단계 탐색 기법은 움직임 벡터가 탐색 영역의 중심에 분포한다는 사실을 이용하여 3단계 탐색 기법을 보안한 기법이다. 육각형 탐색 기법은 탐색 패턴의 중심에 있는 점과 나머지 탐색 점들과의 거리를 일정하게 하여 만든 육각형 탐색 패턴을 사용하여 움직임 벡터를 추정하는 기법이며, 다이아몬드 탐색 기법은 움직임 벡터의 분포의 형태가 다이아몬드라는 사실을 이용하여 만든 다이아몬드 탐색 패턴을 사용하여 움직임 벡터를 추정하는 기법이다.

다이아몬드 탐색 기법은 움직임 벡터가 일반적으로 탐색 영역의 중심에 많이 존재한다는 한다는 것과 움직임 벡터의 분포가 다이아몬드라는 사실을 이용하고 있다. 이 기법은 다음과 같은 탐색 패턴들을 사용하여 움직임 벡터를 추정 한다[5, 8]. 먼저 다이아몬드 탐색 기법은 (그림 2)(a)와 같이 탐색 영역의 원점을 중심으로 8개의 탐색 점들을 배치한 후, 각각의 탐색 점들에 대하여 블록 정합을 수행하여 최소 블록 정합 오차(Minimum Block Distortion : MBD)를 가지고 있는 점을 결정한다. 만약 최소 블록 정합 오차를 가진 점의 위치가 다이아몬드 탐색 패턴의 중심점인 경우, (그림 2)(b)와 같은 작은 다이아몬드 탐색 패턴(Small Diamond Search Pattern)을 사용하여 최소 블록 정합 오차를 가진 점을 결정한다. 이 때 최소 블록 정합 오차를 가진 점의 변위를 움직임 벡터로 추정한다. 그리고 최소 블록 정합 오차를 가진 점의 위치가 다이아몬드 탐색 패턴의 중심점이 아니고 상·하·좌·우 방향에 있을 때는 (그림 2)(c)와 같이 5개의 추가 탐색 점을 배치한다. 만약 최소 블록 정합 오차를 가진 점의 위치가 대각선 방향에 있을 때는 (그림 2)(d)와 같이 3개의 탐색 점들을 추가 배치하여 새로운 다이아몬드 패턴을 만든다. 최소 블록 정합 오차를 가진 점의 위치가 다이아몬드 패턴의 중심점이 될 때까지 위의 과정을 반복 수행하여 움직임 벡터를 추정한다.

일반적으로 다이아몬드와 HEXBS 기법들은 실험에 자주 사용되는 일반적인 영상을 대상으로 움직임 벡터를 찾는데 각각 평균 14회와 11회 정도의 탐색 점을 찾아야만 하는데 반하여 본 논문에서 제안하는 방법은 움직임 벡터를 보다 빠르고 정확하게 추정해나가는 탐색패턴으로 다이아몬드 탐색 기법을 수정한 TDS 알고리즘을 제안하여 기존의 탐색 기법들과 비슷한 화질의 영상을 유지하면서 탐색 점의 수를 대폭 줄였다는데 의미가 있다고 하겠다.



(그림 2) 다이아몬드 탐색 패턴

3. 제안된 탐색 알고리즘 : TDS를 이용한 움직임 추정

기존의 부분 최적 탐색 방법들은 움직임이 작거나 거의 없는 영상에서는 좋은 화질을 유지하는데, 움직임이 많은 영상에서는 움직임 벡터를 찾는데 많은 탐색 점을 사용하고 추정된 움직임 벡터도 정확하지 못한 문제점들을 갖고 있다. 움직임 추정을 하는데 있어서 중요한 관점은 적은 탐색 점수로 움직임 벡터를 찾으면서 좋은 화질을 유지하는 것이다. 따라서 본 논문에서는 화질을 기존 탐색 방법과 유사하게 유지하면서 탐색 점수를 줄이기 위해서 다이아몬드 탐색 기법을 수정하여 세 방향을 고려한 TDS 알고리즘을 제안한다. 이 기법은 움직임 벡터를 보다 빠르고 정확하게 추정해나가는 탐색 패턴이다.

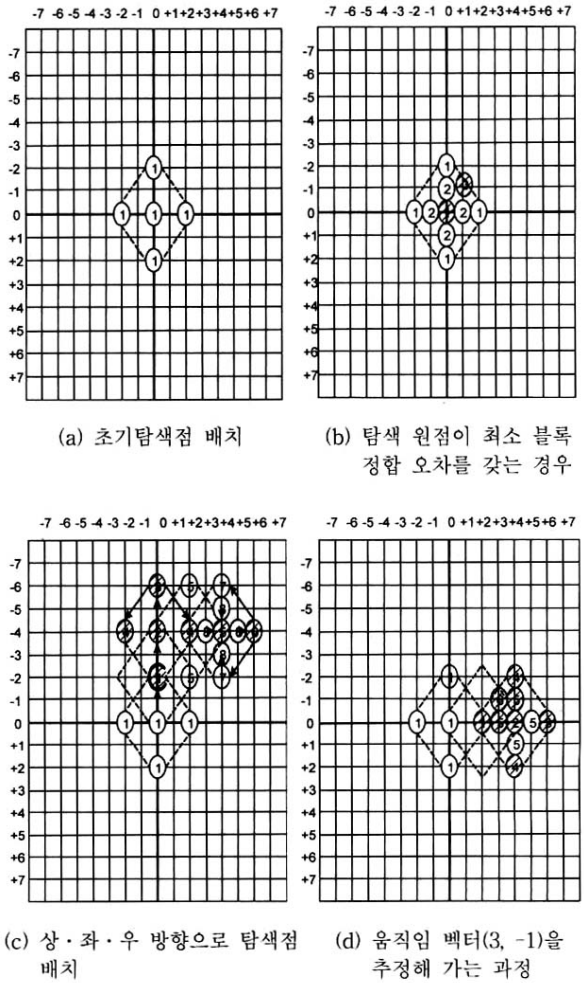
<표 1>의 움직임 벡터의 분포비율과 같이 움직임 벡터의 영상은 대부분 움직임의 방향이 중심에 분포하기 때문에, 탐색 영역의 중심에 실제 움직임 벡터가 집중되어 있다는 특성을 갖는다. <표 1>에서 제시된 통계적 수치를 보면 각 영상의 움직임 벡터가 탐색 원점을 중심으로 거리가 2

이내인 영역의 분포 비율이 움직임이 작은 영상에서는 99% 이상을 차지하고 있고 움직임이 많은 영상에서는 77%를 차지하고 있다. 또한 움직임 벡터의 상·하·좌·우 방향의 비율과 대각선 비율을 보면 움직임이 작은 영상에서는 각각 약 97%와 3%를 차지하고 있고 움직임이 많은 영상에서는 각각 약 65%와 35%를 차지하고 있다. TDS 기법은 이러한 특성을 고려하여 움직임 방향이 대각선 방향보다는 상·하·좌·우 방향의 분포가 크므로 (그림 3)(a)와 같이 탐색 원점을 중심으로 탐색거리를 2로 하여 상·하·좌·우 방향으로 4개의 초기 탐색 점들을 배치한다. 즉, 초기의 탐색 점들은 다이아몬드 기법처럼 8개의 초기 탐색 점을 모두 배치하지 않고 상·하·좌·우 4 방향으로 탐색을 반복하면서 새로운 탐색 원점을 설정한다. 이 기법에서 움직임 벡터를 결정하는 종료 방법은 탐색 기준점이 최소 블록 정합 오차를 갖는 점이라면 작은 다이아몬드 탐색 패턴을 사용하여 (그림 3)(b)에서 ②와 같이 거리가 1인 4개의 추가점을 배치한 후 최소 블록 정합 오차를 실행한다. 만약 최소 블록 정합 오차 가진 점이 탐색 기준점이면 그 점을 움직임 벡터로 결정하고 그렇지 않는 경우 추가의 4점 중에 하나인 경우에는 (그림 3)(b)에서 ③과 같이 최소점과 최소점 좌우측의 2점 중 더 적은 점과의 대각선 거리 1인 탐색 점을 추가하여 블록 정합을 수행한 후 최소블록 정합 오차를 가지면 움직임 벡터로 결정하고[9], 아니면 새로운 4점 중의 한 점을 움직임 벡터로 결정하고 탐색을 종료한다.

〈표 1〉 움직임 벡터의 분포비율

	akiyo	carphone	table
거리가 0인 영역의 분포	97.00%	73.73%	59.42%
거리가 1인 영역의 분포	2.34%	15.70%	17.70%
거리가 2인 영역의 분포	0.44%	4.88%	6.51%
거리가 3인 영역의 분포	0.13%	1.72%	3.37%
십자방향 분포	97.08%	72.52%	65.38%
대각선방향 분포	2.92%	27.48%	34.62%

만약 최소 블록 정합 오차를 가진 점이 탐색 기준점이 아니라 상·하·좌·우 방향에 있는 탐색 점들 중에서새로운 탐색기준점이 선정된다면 (그림 3)(c)에서 ②와 같이 먼저 기준점을 중심으로 수직(상) 방향으로 거리가 2인 1개의 추가 탐색 점을 배치하여 블록 정합을 실행한 후 최소 블록 정합 오차를 갖는 점이 새로 추가한 점이면 (그림 3)(c)에서 ③과 같이 다시 그 점을 기준으로 위의 과정을 반복하면서 탐색을 수행한다. 그렇지 않는 경우, 즉 상 방향으로 추가된 점이 최소 블록 정합 오차가 아니면 (그림 3)(c)에서 ④와 같이 탐색시작점을 기준으로 좌·우 방향으로 2개의 추가 탐색 점을 배치하여 블록 정합을 실행한 후 최소 블록 정합 오차를 갖는 점이 탐색 시작점인 경우 종료 단계를 수행한 후 탐색을 마치고 좌·우 방향으로 2개 중에서 하나가 새로운 탐색 기준점으로 선정되면 다시 수정된 다이아몬드 패턴을 만들기 위해서 (그림 3)(c)에서 ⑤와 같이 3개의 추가 탐색 점을 배치한 후 위의 과정을 반복한다.



(a) 초기탐색점 배치 (b) 탐색 원점이 최소 블록 정합 오차를 갖는 경우

(c) 상·좌·우 방향으로 탐색점 배치 (d) 움직임 벡터(3, -1)을 추정해 가는 과정

(그림 3) TDS 기법

예를 들어 (그림 3)(d)와 같이 움직임 벡터 (3, -1)를 찾아 가는 과정을 살펴보면 다음과 같다. 초기 탐색 점들에서 블록 정합 오차를 실행한 후 최소 블록 정합 오차를 갖는 위치가 탐색 원점이 아닌 (2, 0)라고 하면 그 점을 새로운

탐색 점으로 한다. 그리고 그 점을 중심으로 해서 상 방향으로 거리가 2인 1개의 추가 탐색 점(②)을 배치하여 최소 블록 정합 오차를 구하여 움직임 방향을 예측한다. 만약 움직임이 탐색 기준점을 기준해서 상 방향으로 이동하여 탐색 점 ②의 위치가 최소 블록 정합 오차를 갖는다면, 그 점을 새로운 탐색 기준점으로 하여 상 방향으로 거리가 2인 1개의 추가 탐색 점 ③을 배치한 후 최소 블록 정합을 실행한다. 최소 블록 정합을 실행한 후 최소 블록 정합 오차를 가진 점이 탐색 기준점이다면 더 이상 상 방향으로 탐색하지 않고 좌·우 방향으로 예측 방향을 탐색을 시작한다. 다시 ②를 탐색 기준점으로 하여 좌·우 방향으로 거리가 2인 2개의 탐색 점들(④)을 추가한 후 다시 최소 블록 정합을 실행한다. 최소 블럭 정합 오차가 탐색 기준점이라면 탐색 종료 조건에 의해서 (그림 3)(d)에서 ⑤와 같이 거리가 1인 4개의 추가 탐색 점을 배치한 후 최소 블록 정합을 실행한다. 이 예제의 경우에서는 탐색 기준점이 최소 블록 정합 오차를 갖지 않기 때문에 (그림 3)(d)에서 ⑥과 같이 최소점과 최소점 좌우측의 2점 중 더 적은 점과의 대각선 거리 1인 탐색 점을 추가하여 블록 정합을 수행한다. 추가한 1점이 최소 블록 정합 오차를 가지면 그 점을 움직임 벡터로 결정하므로 최종적인 움직임 벡터는 (3, -1)이 된다.

(그림 4)는 제안한 방법의 순서도이다. 탐색 영역내의 탐색 원점을 중심으로 해서 상·하·좌·우 방향으로 거리가 2인 4개의 초기 탐색 점을 배치하고 블록 정합을 실행한다. 만약 탐색 원점이 최소의 정합 오차를 갖는다면 실험에서 구한 임계 값과 비교한다. 임계 값보다 적으면 그 점을 움직임 벡터로 하고, 임계 값보다 크면 작은 다이아몬드 패턴으로 새로운 탐색 점들을 배치한 후 블록 정합을 실행하여 움직임 벡터를 결정한다. 그렇지 않고 상·하·좌·우 방향에 있는 탐색 점들 중에서 하나가 최소의 정합 오차를 갖는다면 상 방향으로 거리가 2인 1개의 새로운 탐색 점들을 배치하여 블록 정합을 실행한 후 추가된 새로운 탐색 점이 최소 블록정합 오차가 아니면 탐색 기준점을 중심으로 좌·우 2개의 방향으로 거리가 2인 새로운 탐색 점을 추가한 후 최소 블록 정합을 실행하여 상·좌·우 중 하나의 방향을 움직임 방향으로 예측하여 움직임 벡터를 추정한다.

제안된 TDS 알고리즘을 요약하면 다음과 같다.

초기 단계 :

(그림 3)(a)와 탐색 시작점을 기준으로 탐색 거리를 2로 하여 상·하·좌·우 방향으로 4개의 새로운 탐색 점들을 배치한다.

탐색 단계 :

(1 단계) : 배치된 탐색 점들에 대하여 블록 정합을 수행하여 최소 블록 정합 오차를 가진 점을 결정한다. 만약 (그림 3)(b)와 같이 최소 블록 정합 오차를 가진 점의 위치가 탐색 시작점인 경우 종료 단계를 수행한 후 탐색을 마친다. 그렇지 않은 경우에는 (2 단계)를 수행한다.

(2 단계) : 만약 최소 블록 정합 오차를 가진 점의 위치가 상·하·좌·우 탐색 점들 중에 하나가 된다면 (그림 3)(c)와 같이 그 점을 탐색 시작점으로 지정하고, 그 점을 기준으로 수직(상) 방향으로 1개의 추가 탐색 점을 배치하여 블록 정합을 실행한 후 최소 블록 정합 오차를 갖는 점이 새로 추가한 점이면 다시 (2 단계)를 수행하며, 기존 탐색점이면 (3 단계)를 수행한다.

(3 단계) : (그림 3)(d)와 같이 (2단계)의 탐색 시작점을 기준으로 좌·우 방향으로 2개의 추가 탐색 점을 배치하여 블록 정합을 실행한 후 최소 블록 정합 오차를 갖는 점이 탐색 시작점인 경우 종료 단계를 수행한 후 탐색을 마친다. 그렇지 않은 경우(새로 추가한 좌·우 중 한 점이면)에는 최소점인 좌·우 중 한 점을 기준으로 탐색 거리를 2로 하여 상·하·좌·우 방향으로 4개의 새로운 탐색 점들을 배치하고(단, 2단계의 탐색 시작점은 제외) 다시 (1 단계)로 반복 수행한다.

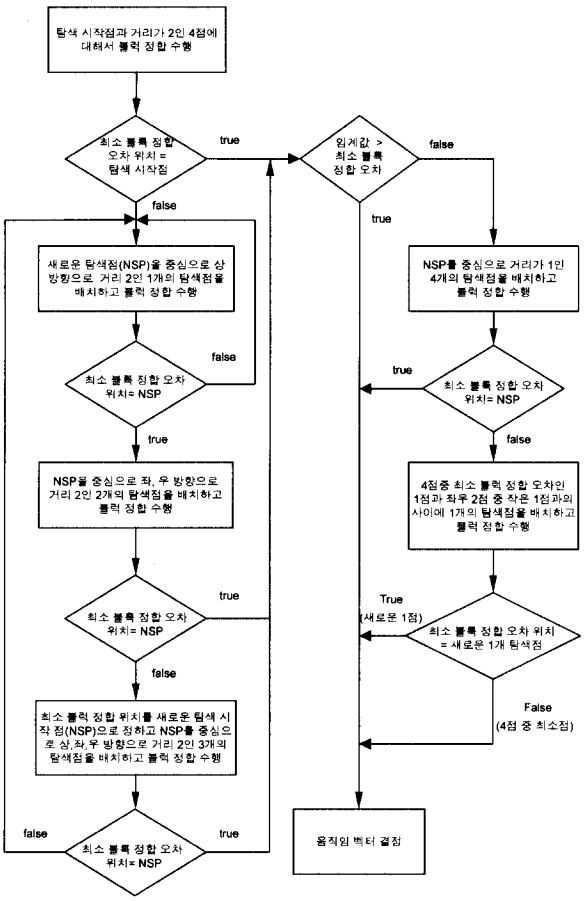
종료 단계 :

(1 단계) : (그림 3)(b)와 같이 초기의 탐색원점이나 (그림 3)(d)와 같이 탐색 과정에서 새로운 탐색 시작점이 최소블록 정합 오차를 갖는다면 먼저 실험에 의한 임계 값과 비교한다. 만약 임계 값 이하의 블록 정합 오차를 갖는다면 그 탐색 원점을 움직임 벡터로 결정하고 종료하고 임계 값 이상이면 (2 단계)를 수행한다.

(2 단계) : 작은 다이아몬드 기법을 이용하여 탐색 거리를 1로 하는 상·하·좌·우 방향으로 4개의 탐색 점들을 배치하여 블록 정합을 수행한다. 최소

블록 정합 오차를 가진 점이 탐색원점이면 그 탐색 원점을 움직임 벡터로 결정한 후 종료하고, 새로운 4점 중의 한 점이면 (3 단계)를 수행한다.

(3 단계) : 최소점과 최소점 좌우측의 2점 중 더 적은 점과의 대각선 거리 1인 탐색 점을 추가하여 블록 정합을 수행한 후 최소블록 정합 오차를 가지면 움직임 벡터로 결정하고, 아니면 새로운 4점 중의 최소점을 움직임 벡터로 결정하고 탐색을 종료한다.



(그림 4) 제안된 방법의 순서도

4. 실험 결과

본 논문에서 제안된 기법의 성능을 평가하기 위하여 실험 영상으로는 QCIF인 Akiyo, Carphone, Claire, Foreman, Mother and Daughter, Salesman, Stefan 그리고 Table 영상의 각각 150 프레임을 사용하였고, 비교 탐색 기법으로는 전역 탐색 기법(FS), 새로운 삼 단계 탐색 기법(NTSS), 다이아몬드 탐색 기법(DS), 그리고 육각형 탐색 기법(HEXBS)들을 사용하였다.

성능 비교 함수로는 영상 화질의 품질을 평가하기 위하여 PSNR(Peak Signal-to-noise Ratio)을 사용하였는데, PSNR은 다음과 같이 정의되었다.

$$MSE = \left(\frac{1}{MN}\right)\sum_{m=1}^{M}\sum_{n=1}^{N}[x(m,n)-\hat{x}(m,n)]^2 \quad (1)$$

$$PSNR = 10\log_{10}\frac{255^2}{MSE} \quad (2)$$

위의 식 (1)에서 M, N은 영상의 가로와 세로의 크기이고, $x(m,n)$는 원 영상화면을 나타내고, $\hat{x}(m,n)$는 복원된 영상화면을 나타낸다. 각 영상에 대한 실험 결과를 <표 2>와 <표 3>에 나타냈는데 <표 2>에는 각 실험 영상에 대한 PSNR(dB)의 평균을 나타내었고, <표 3>에는 각 실험 영상에서 각 블록의 움직임 벡터 추정 시 사용되는 평균 탐색 점수를 나타내었다. <표 2>와 <표 3>에서 나타난 결과를 보면 제안된 기법은 기존의 방법들과 비교할 때 영상 화질 면에서는 유사한 결과를 보이면서 움직임 벡터 추정 시 사용되는 탐색 점수 면에서는 크게 향상을 보였다. 기존의 부분 최적 탐색 기법인 NTSS, DS, 그리고 HEXBS 등의 탐색법과 비교할 때 유사한 화질을 유지하면서도 탐색 점수에서는 평균 38%의 개선된 결과를 얻을 수 있었다. 특히 움직임이 적은 영상에서의 탐색 점수는 50% 향상된 결과를 얻었다. 또한 제안한 TDS와 MVFAST를 비교하면 평균 탐색점수와 영상 화질이 유사하고 특히 빠른 영상 table과 stefan 영상에서는 탐색점수가 적다. 그리고 PMVFAST와 비교하면평균 탐색 점수가 2.25 많으나 화질 면에서는 평균 0.01dB 좋아졌다.

TDS기법은 움직임은 적은 영상에서는 실험에 의한 임계값에 의한 종료 조건에 의해서 탐색 점수를 줄일 수 있었고 움직임이 많은 영상에 대해서는 예측된 방향으로 움직임을 추정하여 움직임 가능성이 낮은 부분을 탐색 대상에서 제외해 나감으로써 탐색이 빠르고 정확하게 한다.

5. 결 론

본 논문은 기존의 탐색 방법 비교해서 움직임 벡터를 보다 빠르고 정확하게 추정해 나가기 위해서 탐색 점수를 크게 줄이면서 영상의 화질은 유사하게 하는 3 방향을 고려

한 TDS 알고리즘을 제안하였다. 이 기법에서는 움직임 벡터가 탐색 영역 중심점을 기준으로 대각선 방향보다는 상·하·좌·우 영역에 집중되어 있는 특성을 이용하여 먼저 탐색 원점을 중심으로 상·하·좌·우 4 방향으로 탐색 점을 배치 한 후 블록 정합을 실행한다. 이들 중 정합 오차가 가장 작은 지점을 기준으로 상·좌·우 3 방향으로 탐색 점을 확장하여 정합 오차를 측정해 보고 3 방향 중 한 방향을 움직임 방향으로 예측한다. 예측된 방향으로 위의 과정을 반복하며 움직임을 추정한다. 이 기법을 이용하면 움직임 가능성이 낮은 부분을 탐색 대상에서 제외해 나감으로써 탐색 점수를 크게 줄일 수 있어 탐색이 빠르고 정확하게 이루어지고 영상의 화질 면에서 유사함을 보였다.

〈표 2〉 PSNR 결과

	FS	NTSS	DS	HEXBS	MVFAST [10]	PMVFAST [11]	제안방법 (TDS)
akiyo	35.42	35.42	35.42	35.43	35.40	35.42	35.42
salesman	33.72	33.71	33.72	33.71	33.71	33.72	33.73
claire	35.69	35.65	35.58	35.45	35.52	35.43	35.56
mother	32.49	32.47	32.48	32.47	32.43	32.43	32.46
foreman	30.98	30.79	30.73	30.62	30.62	30.62	30.66
carphone	32.22	32.19	32.17	32.14	32.08	32.07	32.11
table	31.31	31.28	31.23	31.21	31.25	30.95	31.20
stefan	28.10	28.05	28.06	28.07	28.09	28.04	28.04

〈표 3〉 탐색 점수 결과

	FS	NTSS	DS	HEXBS	MVFAST [10]	PMVFAST [11]	제안방법 (TDS)
akiyo	961	17.04	13.01	11.00	5.24	2.9	6.62
salesman	961	17.12	13.07	11.04	5.33	5.3	6.87
claire	961	17.09	13.09	11.05	4.80	4.5	6.78
mother	961	17.32	13.24	11.11	8.12	4.7	7.85
foreman	961	19.16	14.80	11.80	11.18	7.8	9.98
carphone	961	18.14	13.98	11.45	10.06	7.7	9.15
table	961	19.12	15.08	12.24	11.44	8.5	10.21
stefan	961	20.44	16.07	12.80	11.78	8.9	10.64

참 고 문 헌

[1] J. Y. Tham, S. Ranganath, A. A. Kassim, "A Novel Unrestricted Center-Biased Diamond Search Algorithm for Block Motion Estimation," IEEE Transactions on Circuits and Systems for Video Technology, Vol.8, No.4, pp.369-377, Aug., 1998.

[2] Z. Shan, M. Kai-kuang, "A New Diamond Search Algorithm for Fast Block Matching Motion Estimation," IEEE Transactions on Image Processing, Vol.9, No.2, pp. 287-290, 2000.

[3] T. Koga, K. Iinuma, Y. Hirano, Y. Iijim, T. Ishiguro, "Motion-compensated interframe coding for video conference," In Proc. Nat. Telecommun. Conf., pp.G5.3.1-G5.3.5, Nov., 1981.

[4] L. Renxiang, Z. Bing, M. L. Liou, "A New Three Step Search Algorithm for Block Motion Estimation," IEEE Transactions on Circuits and Systems for Video Technology, Vol.4, No.4, pp.438-442, 1994.

[5] P. Lai-Man., M. Wing-Chung, "A Novel Four-Step Search Algorithm for Fast Block Motion Estimation," IEEE Transactions on Circuits and Systems for Video Technology, Vol.6, No.3, pp.313-317, 1996.

[6] C. Yuk-Ying, W. B. Neil, "Fast search block-matching motion estimation algorithm using FPGA," Visual Communication and Image Processing 2000, Proc. SPIE, Vol. 4067, pp.913-922, 2000.

[7] Ce Zhu, Xiao Lin and Lap-Pui Chau, "Hexagon-Bassed Search Pattern for Fast Block Motion Estimation," IEEE Transaction on Cuircuits and Systems for Video Technology, Vol.12, No.5, pp.349-355, May, 2002.

[8] C. Guy, G. Michael, K Faouzi, "Efficient Motion Vector Estimation and Coding for H.263-based very low bit rate video compression," ITU-T SG 16, Q15-A-45, 1987.

[9] 이근영, "5-방향 탐색 알고리즘을 이용한 고속 움직임 벡터 예측", 전자공학회 논문지, 제5권 S편 제3호, pp.448-453, Mar., 1998.

[10] Prabhudev Irappa Hosur, Kai-Kuang Ma, "Report on Performance of Fast Motion using Motion Vector Field Adaptive Search Technique," ISO/IEC/ JTC1/SC29/WG11 M5453, Maui, December, 1999.

[11] "Optimization Model Version 1.0," ISO/IEC JTC1/SC29/WG11 MPEG2000/N3324, Noordwijkerhout, Nethelands, Mar'00.

김 미 영

e-mail : kimmee@namdo.ac.kr

1983년 전남대학교 계산통계학과 이학사

1985년 이화여자대학교 대학원 이학석사

1997년 전남대학교 대학원 전산통계학과 이학박사

1986년~1997년 목포과학대학 전자계산과 부교수

1998년~현재 전남도립남도대학 컴퓨터정보통신과 교수

관심분야 : CAD/VLSI, 멀티미디어 등

정 미 경

e-mail : mgjung@chonnam.chonnam.ac.kr

1987년 전남대학교 전산통계학과(이학사)

1989년 전남대학교 대학원 전산통계학과 이학석사

1995년~현재 전남대학교 대학원 전산통계학과 이학박사

관심분야 : VLSI/CAD, 인공지능, 멀티미디어