공 학 박 사 학 위 논 문

비디오 압축에서 PDE기반 고속 움직임 추정기법에 관한 연구

2008년 2월

부 경 대 학 교 대 학 원

전 자 공 학 과

류 태 경

공 학 박 사 학 위 논 문

비디오 압축에서 PDE기반 고속 움직임 추정기법에 관한 연구

지도교수 문 광 석

이 논문을 공학박사 학위논문으로 제출함

2008년 2월

부 경 대 학 교 대 학 원

전 자 공 학 과

류 태 경

류태경의 공학박사 학위논문을 인준함

2008년 2월 26일

주 심 공학박사 권 태 하 인

위 원 공학박사 권 기 룡 인

위 원 공학박사 김 종 남 인

위 원 공학박사 류 권 열 인

위 원 공학박사 문 광 석 인

목 차

그림 차례

도표 차례

A Study on Fast Motion Estimation Techniques Based on Partial Distortion Elimination in Video Coding

Tae Kyung Ryu

Department of Electronic Engineering, The Graduate School,

Pukyong National University

Abstract

Video compression is an essential tool for video data storage and visual communication. The basic principle of video compression is to remove data redundancies, including spatial, temporal, statistical, and spectral redundancies. Motion estimation is an indispensable technique to reduce the temporal redundancy. In the motion estimation techniques for video sequence compression, the block matching method is the most popular in current international video coding standards such H.261, H.263 and MPEG-1/2/4 due to its simplicity and good error performance.

Full search(FS) in the block-matching algorithm(BMA) based on the translational motion model finds the location with the minimum value of matching errors of all candidate displacements within a given search range. It has been popularly used in video coding applications because of its simplicity and easy hardware implementation. However, its heavy computational load for a large search range can be a serious problem in real-time video coding applications. Many fast motion estimation algorithms to reduce the computation of the full search have been studied in the last decade. We can classify these fast motion estimation methods into two main groups: The former is lossy

motion estimation group has degradation of predicted images compared with the conventional FS, and the latter is lossless motion estimation group has no any degradation of predicted images compared with the conventional FS algorithm. A big problem of the lossy fast motion estimation algorithms can be falling into a local minimum, which is not optimal motion vector. The remaining limitation of the lossless fast motion estimation algorithms is low computational reduction ratio compared with the lossy ones.

In this thesis, we propose fast motion estimation algorithms based on partial distortion elimination(PDE) to improve computational reduction ratio and speedup computational time of conventional PDE algorithm. Our algorithms are composed of three parts. The first algorithm identifies computational matching order from initial calculation of matching differences. According to the computational order identified, matching errors are calculated based on partial distortion elimination method. The proposed algorithm takes about 8~20% of computations for block matching error compared with conventional FS algorithm. The second algorithm is based on selective matching scan and elimination of unlike candidate blocks from initial matching error. According to the obtained matching order, we proceed to calculate matching errors and remove unlike candidate vectors based on PDE method. Our algorithm takes about 4~10% of computations for block matching error compared with conventional FS algorithm. The third algorithm is based on the prediction of threshold of sub-blocks regarding with the information of 3 neighbors block adaptively. We proposed a new block matching algorithm by sorting square sub-block and applying refining

adaptive threshold for each block while keeping the same prediction quality compared with the original FS algorithm. Our algorithm takes about 2~5% of computations for block matching error compared with conventional FS algorithm. If we cascade the successive elimination algorithm(SEA), multi-level successive elimination algorithm(MSEA) or hexagonal based search(HEXBS) algorithms to our proposed algorithm, we can further remove only unnecessary computation. Thus our algorithm will be useful to real-time video coding applications using MPEG-4 AVC or MPEG-2 video coding standards.

제 1 장 서 론

최근 컴퓨터의 발전 및 디스플레이 기기의 발전으로 인해 멀티미디어 데이터의 사용이 일반화 되면서 그 가운데 대부분의 정보를 차지하는 동영상 데이터의 이용이 보편화 되고 있다. 이러한 동영상 데이터는 많은 데이터를 포함하고 있으며 압축 없이 전송할 경우 많은 데이터량으로 인해 저장 및 전송에 있어 심각한 문제를 가지고 있다. 이러한 문제점을 해결하기 위해 저장 용량을 줄이고 전송시간을 짧게 하기 위해 중복된 데이터를 줄이는 기술을 영상 압축이라고 한다. 영상 압축은 자료의 양을 축소하여, 보다 효과적으로 저장하고 전송할 수 있게 한다. 영상압축에 있어서 영상 부호화의 목적은 적은 양의 정보로 원 영상을 충실히 표현하고 재생하는 데 있다. 일반적으로 영상의 중복성은 시간적 중복성, 공간적 중복성, 통계적 중복성을 가지는데 시간에 따라 변화하는 동영상 시퀀스는 시간 상관계수가 공간 상관계수보다 훨씬 크기 때문에 공간 상관성을 이용하는 프레임내의 부호화 보다 프레임간의 높은 상관성을 이용하여 프레임간의 시간적 중복성을 감소시켜 압축효율을 증가시키는 방법을 사용한다.

움직임 추정(motion estimation)은 시간적 중복성을 줄이기 위해 이전 프레임과 현재 프레임간의 관계를 이용하여 움직임 벡터(motion vector)를 구해 비디오 데이터를 압축 하고, 원영상과 보상된 영상의 차이를 부호화 하게 되는데 두 영상의 차이가 클수록 데이터량이 증가하므로 최적의 움직임 벡터를 찾는 움직임 추정은 압축의 중요한 역할을 한다. 이 방법은 H.261, H.263, MPEG-1, MPEG-2 MPEG-4/AVC등의 동영상 압축 표준에 사용되고 있다

[1]-[26]. 블록 매칭 방법은 좋은 예측 성능, 작은 계산 로드, 그리고 작은 움직임 파라미터 정보 등에 기인한다. 블록 기반 매칭 방법은 참조블록과 후보블록 간의 매칭 에러를 최소화 하는 최적의 움직임 벡터를 찾는 알고리즘이다.

블록 매칭 방법에서 가장 대표적인 전역 탐색 방법은 주어진 탐색 영역에서 최소의 매칭 에러를 갖는 후보 지점을 찾는 것이다. 이 탐색 방법은 간단하고 쉬운 하드웨어 구현 때문에 비디오 데이터 부호화에서 널리 사용되어져 왔다[27-39]. 동영상 압축 표준들의 부호화 과정은 움직임 추정, 움직임 보상, DCT(discrete cosine transform), 양자화, 엔트로피 부호화 등으로 구성되는데 H.261의 경우 움직임 추정은 탐색영역이 ±7일 경우 전체 부호화 계산량의 60% 정도를 차지하고, ±128의 탐색영역을 가지는 HDTV의 경우는 전체 계산량의 90%이상을 차지한다. 이 방법의 방대한 계산량은 실시간 부호화 응용 분야에서 심각한 문제점으로 남겨져 왔다. 이러한 전역 탐색방법의 계산량을 줄이기 위해 많은 고속 알고리즘들이 연구 되어져 왔는데, 이들 고속 알고리즘들은 크게 두 그룹으로 나누어 질 수 있다. 하나는 전역 탐색 방법에 비해 예측화질의 손실을 감수 하더라도 계산량 감소에 중점을 두어 계산량 감소를 이루는 손실 움직임 추정 방법이고, 다른 하나는 전역 탐색 방법에 비해 예측 화질의 손실 없이 계산량의 감소를 이루는 무손실 움직임 추정 방법이다. 손실 움직임 추정 기법은 다음의 세부 그룹으로 다시 나누어 질 수 있다. 단일 에러 표면 가정(unimodal error surface assumption-UESA) 기법[40]-[46], 다해상도(multi-resolution) 기법[46]-[55], 움직임 벡터의 시/공간 상관관계를 이용한 탐색 영역 기법, 매칭 에러의 문턱값을 이용한 중간 멈춤 기법, 매칭 블록의 행/열의 투영(integral projection technique) 기법, 더 낮은 비트 해상도(low bit resolution) 기법, 매칭 블록의 서브샘플링(subsampling) 기법 등이 있다[40]-[72]. 무손실

움직임 추정 기법에는 다음과 같은 것들이 있다. 기준 블록과 후보 블록 그리고 그때까지의 최소 에러값을 이용한 연속 후보 제거 알고리즘(successive elimination algorithm-SEA) 및 그 것의 변형된 알고리즘들 [73-74], 기준 블록과 후보 블록에 대해 수직, 수평, 블록 전체의 투영(vertical, horizontal and massive projection)을 이용한 알고리즘[46-48], 그리고 부분 에러 제거(partial distortion elimination-PDE) 알고리즘 및 그 변형 알고리즘 [75]-[89] 등이 있다. 손실 움직임추정은 예측 알고리즘의 대부분은 지역 최소화 문제로 인한 정확하지 못한 움직임 벡터 때문에 예측화질의 열화를 초래한다. 특히 비트율이 제한된 응용분야에서 고속 알고리즘으로 인한 부정확한 움직임 벡터는 증가된 예측 에러로 인해 심각한 문제를 야기 시킬 수 있다. 최근의 손실 움직임 추정 방법의 경우 계산량을 감소와 동시에 화질의 열화를 줄이는 쪽으로 연구가 진행되고 있다. 무손실 움직임 추정의 대표적인 SEA는 기준 블록의 합과 후보 블록의 합 그리고 그때까지의 최소 매칭 에러를 이용하여 후보 지점들을 제거하는 방법이다. 이 방법은 기준 블록의 합과 후보블록의 합의 중복 계산을 줄이기 위해 행의 합 또는 열의 합을 미리 계산한다. MSEA의 경우는 보다 많은 후보블록을 제거하기 위해 기준 블록의 합과 후보 블록의 합을 보다 작게 나누어 그때까지의 최소 매칭 에러와 비교한다. SEA기반 방법들에서 후보 지점을 미리 제거하기 위한 기준 블록의 합과 후보 블록의 합은 보다 계산량 감소에 제한을 가지게 한다. 또한 최소 매칭에러의 크기에 따라 SEA방법의 효율이 결정된다. PDE 알고리즘은 후보블록의 에러값에 대한 완전한 계산을 하기 전에 부분 매칭 에러를 가지고 최소 에러와 비교함으로써 불필요한 계산을 줄이는 방법을 사용 한다. 매칭에러의 중간 합이 그때까지의 최소 에러보다 크다면 나머지 계산을 할 필요가 없다는 것이다. 이 방법은 전역 탐색에 비해 예측화질

의 손실이 발생하지 않으면서 보다 빠른 움직임 벡터를 찾을 수 있지만 후보 지점을 줄여서 움직임 벡터를 찾는 손실 움직임 추정 방법에 비해 계산량 감소는 제한적이다. PDE 알고리즘에서 보다 빠르게 후보블록을 제거하기 위해서는 최소 에러값을 갖는 블록이 스캔 시에 우선적으로 나오게 하는 방법과 부분 에러값이 큰 서브블록을 우선적으로 비교할 수 있는 매칭방법에 관한 연구가 필요하다. 그리고 PDE 알고리즘의 중요한 특징은 매칭 에러값을 비교하는 과정에 적용하기 때문에 다른 고속 움직임 추정 방법들에 적용하여 사용할 수 있다.

제안한 방법은 이러한 PDE 알고리즘의 계산량을 줄여 보다 빠른 움직임 추정을 위한 세 가지 방법을 제안한다. 첫 번째 방법은 최소 블록 매칭 에러 계산 시 서브 블록별 복잡도를 계산하여 에러값이 큰 서브블록이 우선적으로 매칭 될 수 있도록 매칭 순서를 결정하는 방법을 제안한다. 제안한 방법은 각 매칭 블록에 서브블록의 매칭순서를 적응적으로 결정하여 보다 빠르게 불필요한 후보블록을 제거할 수 있다. 두 번째 방법은 보다 빠르게 후보블록을 제거하기 위하여 최초 서브블록 매칭 시에 최초 서브블록의 에러값과 최소 에러값에 임계치를 적용한 값을 비교하여 서브블록의 에러값이 최소 에러값의 임계치를 적용한 값보다 크다면 후보블록에서 제외하여 이후 블록 매칭을 제거하여 보다 빠르게 불필요한 후보블록을 제거하였다. 세 번째 방법은 전역 탐색 방법과 비교하여 예측화질의 손실이 발생하지만 보다 빠르게 후보블록을 제거하기 위하여 각 서브블록의 에러값 계산 시 서브블록의 크기에 최소 에러값을 적응적으로 추정하여 보다 빠르게 후보블록을 제거하는 움직임 추정 방법을 제안한다. 기존의 방법의 경우 임계치 추정 시 전역 탐색 방법과 비교할 때 최소 에러값을 갖는 후보블록의 오검출로 인한 화질의 열화를 발생의 문제를 제안한

방법은 현재 후보블록과 인접 후보블록간의 유사도에 기초하여 적응적으로 예측하여 화질의 열화가 거의 발생하지 않으면서 보다 빠르게 최적의 움직임 벡터를 찾을 수 있었다. 제안한 방법들 역시 PDE기반의 방법들로 기존의 고속 움직임 추정방법들에 적용 할 수 있다. 제안된 첫 번째와 두 번째 방법을 기존의 SEA와 MSEA에 적용하여 본 결과 보다 빠르게 움직임 벡터를 찾을 수 있었다. 세 번째 방법은 전역 탐색에 비해 화질의 열화가 발생할 수 있으므로 기존의 고속 손실 압축방법 가운데 가장 우수한 성능을 가지는 HEXBS(hexagon based search)방법에 적용하여 보다 많은 계산량의 감소를 이루었다.

본 논문의 구성은 다음과 같은 순서로 구성된다. 제 2장에서는 움직임 예측의 개념과 블록 매칭 방법에 대해서 설명할 것이다. 그리고 블록 매칭 방법 가운데 최적의 성능을 가지는 전역 탐색 방법에 대하여 알아본다. 그리고 이 방법을 개선한 고속 움직임 예측 방법에 대하여 설명하고 고속 움직임 예측 알고리즘을 분류해 본다. 제 3장에서는 제안한 알고리즘의 동기 및 연구 그리고 본 논문의 움직임 예측을 적용한 알고리즘을 설명한다. 제 4장에서는 제안한 세 가지 방법을 여러 실험영상에 적용하여 나타난 결과를 비교하고 결과의 성능을 평가한다. 마지막으로 제 5장에서는 결론을 맺는다.

제 2 장 움직임 추정 알고리즘

2.1 움직임 추정

2.1.1 비디오 압축

영상압축에 있어서 영상 부호화의 목적은 적은 양의 정보로 원 영상을 충실히 표현하고 재생하는 데 있다. 일반적으로 영상의 중복성은 시간적 중복성, 공간적 중복성, 통계적 중복성을 가지는데 시간에 따라 변화하는 동영상 시퀀스는 시간 상관계수가 공간 상관계수보다 훨씬 크기 때문에 공간 상관성을 이용하는 프레임내의 부호화 보다 프레임간의 높은 상관성을 이용하여 프레임간의 시간적 중복성을 감소시켜 압축효율을 증가시키는 방법을 사용한다. 표 1은 영상의 중복성과 MPEG이 중복성을 줄이는 방법을 나타낸다.

표 1. 영상의 중복성과 MPEG에서의 응용

Redundancy	MPEG에서의 응용
시간적 중복성 (Temporal Redundancy)	움직임 추정(Motion Estimation)
공간적중복성 (Spatial Redundancy)	DCT and Quantization DCT계수와 움직임 벡터의 예측코딩
통계적 중복성 (Statistical Redundancy)	VLC(Variable Length Code)

동영상에서 시간적 중복성을 줄이기 위하여 이전 프레임과 현재 프레임간의 관계를 움직임 벡터(motion vector)를 구해 이용하는 움직임 추정(motion estimation)은 비디오 데이터를 압축 하는데 적합한 기술이며, 추정된 움직임 벡터로 보상된 영상과 원영상의 차이가 영상의 데이터량을 나타내는 중요한 역할을 한다. 그림 2.1은 영상 압축 인코더의 움직임 벡터 인코딩 과정을 나타내고 있다. 움직임 예측 및 보상은 인코더에서 이전 프레임과 현재프레임에서 움직임을 검출하여 움직임 벡터와 프레임간의 오차값을 인코딩하여 낮은 비트로 압축을 구현할 수 있다. 그러므로 움직임 추정과 보상을 사용하는 변환 코딩은 비디오 압축 표준인 H.261, H.263, MPEG-1, MPEG-2 MPEG-4/AVC등에서 사용되고 있다[90]-[93].

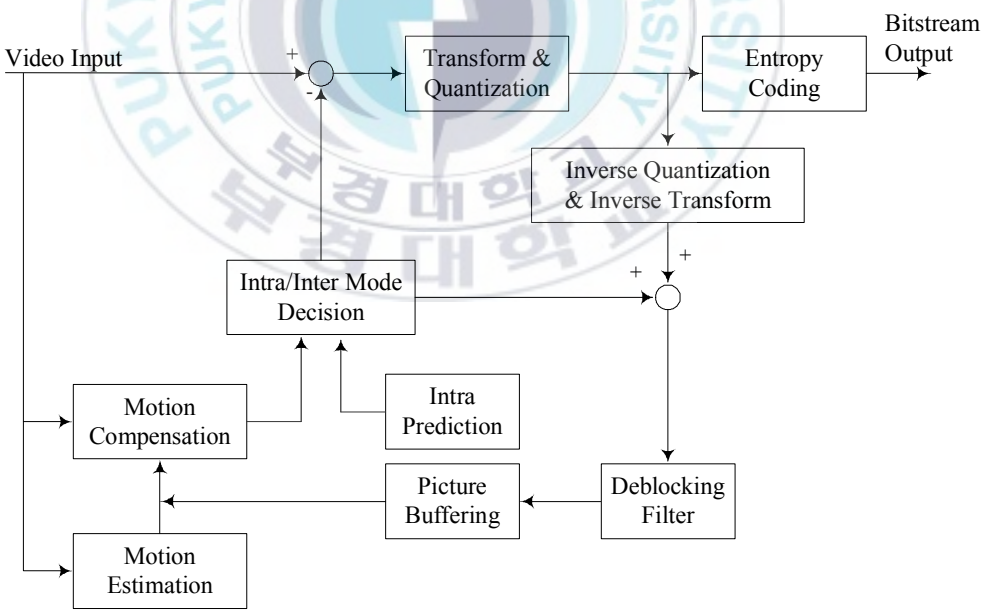

그림 2.1 영상 압축 인코더

일반적으로 움직임 추정은 세 가지 방법으로 구분할 수 있다. 블록매칭 방법, pel-recursive 방법, optical flow방법 등이 있다. 이들 가운데 블록 매칭 방법만이 사용되는데, 좋은 예측 성능, 작은 계산 로드, 그리고 작은 움직임 파라미터 정보 등에 기인한다. 블록 기반 매칭 알고리즘은 참조블록과 후보블록간의 매칭 오차를 최소화 하는 최적의 움직임 벡터를 찾는 알고리즘이다. 그러므로 pel-recursive 방법과는 달리 블록안의 모든 픽셀에 동일한 움직임 벡터가 사용되어진다.

2.1.2 블록 매칭 알고리즘(block matching algorithm)

시간적인 중복성은 영상 부호화의 압축 효율에 중요한 역할을 한다. 그러므로 대부분의 부호화 방법들이 압축방식에 있어서 복호기로 이미 전송된 영상들을 활용하고 있다. 움직임은 연속된 영상들 사이의 차이를 만드는 주요 원인이 되기 때문에 움직임 보상 기술은 다음 영상을 예측하는데 적용되고 따라서 움직임 추정이 이루어져야 한다. 때때로 움직임 추정은 부호기와 복호기에서 동시에 이루어질 수 있는데 부호기에서 주로 이루어지고 복호기로는 움직임 정보가 전송된다. 적절한 움직임 모델의 선택은 응용 모델에 따라 다르기 때문에 그 다양성이 고려되어야 한다. 그러나 오늘날 비디오 압축 표준에서는 블록마다 하나의 움직임 벡터가 전송되고 블록의 분할에 따라서 적응적인 움직임 모델을 사용한다. MPEG-4와 H.26L 표준에서도 낮은 부호화 율을 위해서 움직임 모델을 사용하고 있다. 블록의 분할에 따른 적응적인 움직임 모델은 그 단순성에 따라서 높은 효율을 가진다. 그림 2.2에서와 같이 움직임 추정에 있어서 각 영역마다 적절한 움직임 벡터를 찾는 문제는 대부분의 경우에 이전 영상의 탐

색 영역에서 최적의 매칭 영역을 탐색하는 것이다. 즉 움직임 추정은 곧 움직임 탐색이라고 할 수 있다.

움직임 추정에서 연산량의 대부분은 블록 매칭을 위한 왜곡의 계산과정에서 나온다. 자주 사용되는 매칭기준에는 SAD(sum of absolute difference), MAD(mean absolute difference) 와 MSE(mean-square error)등이 있는데 이 같은 매칭 기준들은 픽셀기준으로 계산하기 때문에 계산의 강도가 높다. 어떤 방법들은 블록의 전체 왜곡을 픽셀 부분 왜곡으로 대체하여 왜곡의 계산을 단순화하기도 한다. 이는 약간의 성능 손실이 있지만 계산량을 반으로 줄이는 장점이 있다.

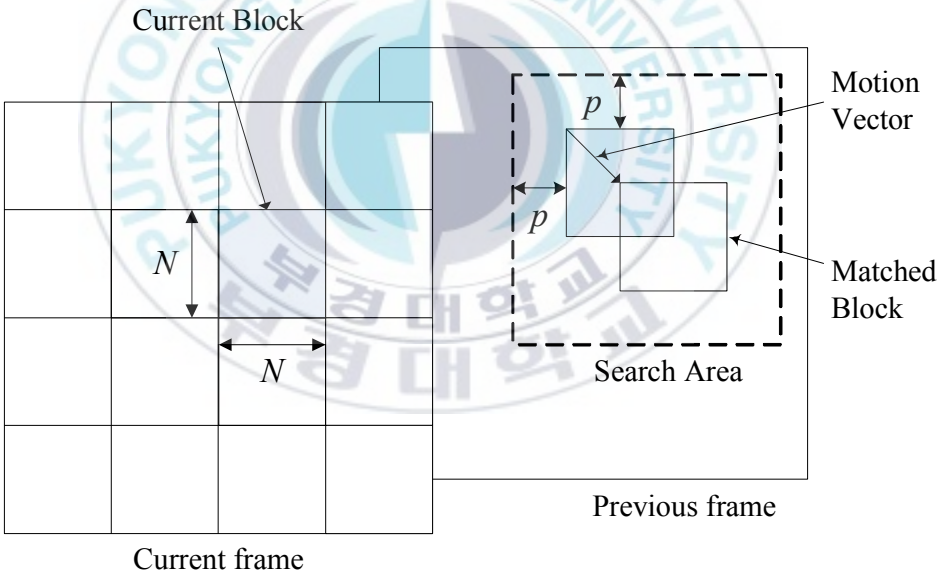

그림 2.2 움직임 추정을 위한 블록 매칭

2.2. 전역 탐색 알고리즘과 고속 블록 매칭 알고리즘

2.2.1 전역 탐색 알고리즘

전역 탐색방법은 탐색영역 내 모든 후보 블록들을 체크한 후 매칭 에러가 가장 작은 후보블록을 최선의 매칭 점으로 선택한다. 오차값을 측정하는 많은 방법들이 제안되었고 이들 가운데 블록매칭에러(sum of absolute difference-SAD)가 가장 많이 쓰인다. $N \times N$의 크기를 갖는 MB(Macro Block)에서 $SAD(x,y)$는 식(2-1)과 같다.

$$SAD(x,y) = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} |f_n(i,j) - f_{n-1}(i+x, j+y)| \tag{2-1}$$

식(2-1)에서 $f_n(i,j)$는 현재 MB의 픽셀(i,j)값을 나타내고 $f_{n-1}(i+x, j+y)$는 이전 프레임의 후보블록의 픽셀 값을 나타낸다. 움직임 벡터를 찾는 가장 간단한 방법은 탐색 영역내의 모든 위치에 대해 SAD를 구하고 SAD의 최소값을 가지는 위치를 움직임 벡터로 결정하는 것이다. 이 방법을 전역 탐색 방법이라고 한다. 이 방법은 많은 계산량을 요구하지만 SAD에 대하여 최적의 움직임 벡터를 찾을 수 있다는 큰 장점이 있다.

전역 탐색방법은 계산량의 측면에서는 과도하지만, 최소 SAD를 가지는 움직임 벡터를 항상 찾을 수 있다는 장점이 있다. 전역 탐색의 많은 계산량 때문에 전역 탐색을 대체 할 수 있는 고속의 움직임 추정 방법들이 나타나게 되었다. 이러한 방법들은 전역 탐색방법과 비교하여 계산량을 크게 감소시키면서 준 최적(sub-optimum)의 성능을 나타내게 되었다. 여기서 준 최적이라 함은

이러한 방법들이 최소 SAD를 가지는 움직임 벡터를 찾을 수 있음을 보장하지 않는다는 말과 동일하다.

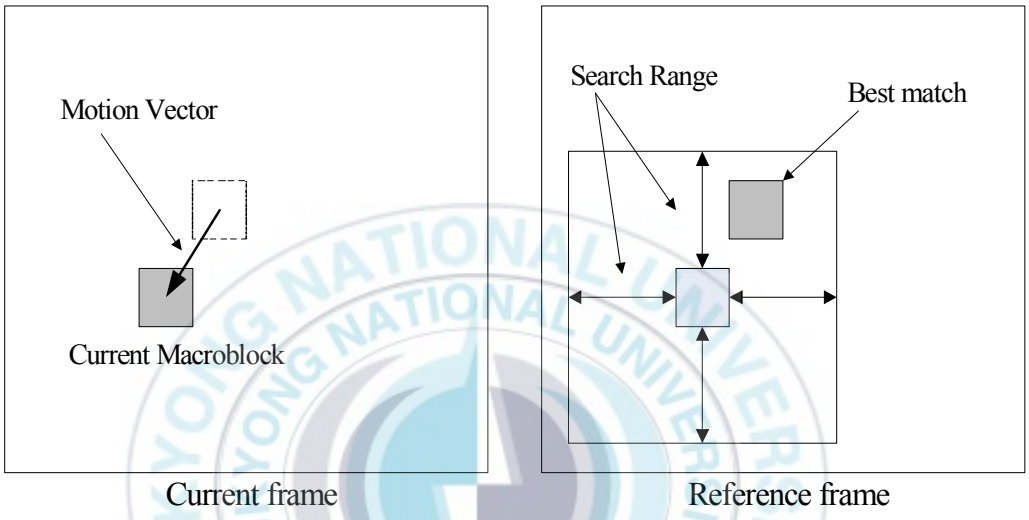

그림 2.3 현재 블록과 같은 중심을 가지는 이전영상의 탐색영역에 대한 전역 탐색의 예

2.2.2 고속 움직임 추정 알고리즘

전역 탐색방법의 계산상의 복잡성으로 인해 고속 움직임 추정 방법들이 제안되었다. 이들 방법가운데 후보벡터의 수를 줄여 계산량을 줄이는 방법들이 제안되었는데 이들의 구현의 용이함 및 계산량 감소는 아주 매력적이다. 하지만, 이 방법들은 SAD함수가 최적 매칭 위치로부터 멀어짐에 따라 단조 증가(monotonically increasing)하는 가정이 성립할 경우 우수한 성능을 보인다. 그러나 이러한 가정이 제대로 맞지 않을 경우 전역 최소값보다 지역 최소값으로 빠질 수 있다.

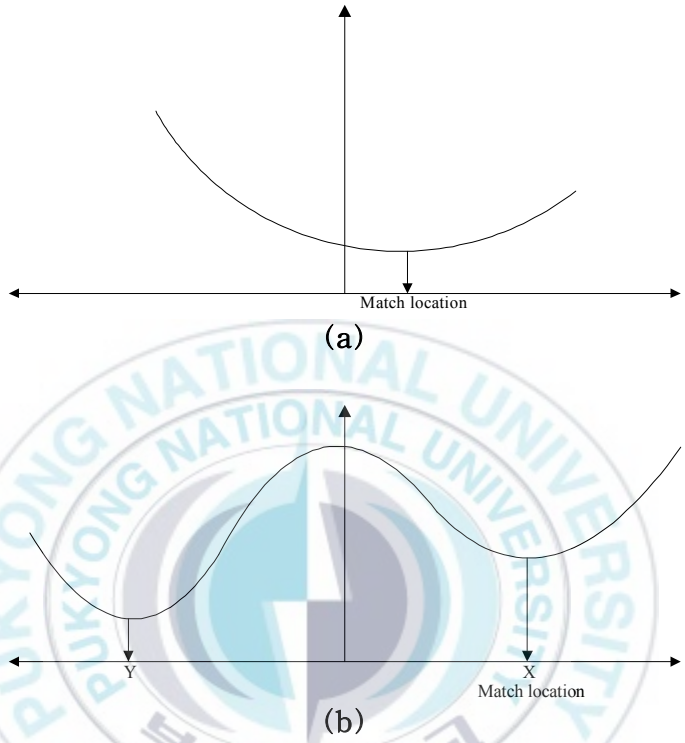

그림 2.4 탐색 위치에 따른 SAD 함수의 예
(a)단조 증가함수 (b)단조 증가함수가 아닌 경우

그림 2.4는 이러한 예를 보여주고 있다. 그림 2.4(a)는 하나의 전역 최소값을 가지는 경우이다. 그래서 고속 탐색법의 결과가 최적의 결과와 일치하게 된다. 여기서 최적의 결과라고 함은 전역 탐색방법으로 찾아진 움직임 벡터를 말한다. 그림 2.4(b)는 하나의 전역 최소값과 하나의 지역 최소값이 존재하는 예이다. 이 경우, 고속 탐색법이 최소 SAD를 가지는 위치를 Y대신 X로 판단하는 경우가 발생할 수 있다. 그래서 지역 최소값에 빠질 수 있는 고속 탐색법의 경우 전역 탐색방법에 비해 보다 큰 움직임 추정 에러를 가질 수 있다.

이들 고속 알고리즘들은 크게 두 그룹으로 나누어 질 수 있다. 하나는 전역 탐색 방법에 비해 예측 화질의 손실을 갖는 것이고, 다른 하나는 예측 화질의 손실을 갖지 않는 방법이다. 전자는 다음의 세부 그룹으로 다시 나누어 질 수 있다. 단일 에러 표면 가정(unimodal error surface assumption-UESA) 기법, 다 해상도(multiresolution) 기법, 움직임 벡터의 시/공간 상관관계를 이용한 가변 탐색 영역 기법, 매칭 에러의 문턱값을 이용한 중간 멈춤 기법, 매칭 블록의 행/열의 투영 (integral projection technique) 기법, 더 낮은 비트 해상도 (low bit resolution) 기법, 매칭 블록의 서브샘플링(subsampling) 기법 등이 있다. 그리고 후자의 무손실 움직임 예측 기법에는 다음과 같은 것들이 있다. 기준 블록과 후보 블록의 블록합을 이용한 후보 제거 알고리즘(successive elimination algorithm-SEA) 및 그 것의 변형된 알고리즘들, 고속의 2차원 FIR필터링 방법을 이용한 고속 알고리즘, 기준 블록과 후보 블록에 대해 수직, 수평, 블록 전체의 투영(vertical, horizontal and massive projection)을 이용한 알고리즘, 그리고 부분 매칭 에러값을 이용한 후보 제거 (partial distortion elimination-PDE) 알고리즘 및 그 변형 알고리즘 등이 있다.

2.3 고속 손실 블록 매칭 알고리즘

2.3.1 3단계 탐색(three-step search) 방법

3단계 탐색 기법은 넓은 영역에 걸쳐 몇 개의 탐색 후보 점을 조사한 후, 점차 범위를 좁혀 나가는 방식으로 가장 간단하면서도 효율적인 알고리즘이다. 3단계 탐색 기법은 탐색영역의 중심으로부터 4화소 간격의 9×9의 9개의 탐색 후보 점을 가진 사각형 블록 형태로 분포된 탐색 후보 점들로 구성하여 한 개의 매크로 블록의 총 25개의 탐색 후보 점에 대해 최소의 매칭 기준 에러값에 의해 3단계로 나누어 움직임 벡터를 탐색한다. 그러나 3단계 탐색 기법은 계산량을 줄이고 구현이 간단하다는 장점은 가지고 있으나 모든 영상에 대해서 고정된 탐색패턴을 적용하기 때문에 움직임이 거의 없는 정적 블록인 경우는 문제가 없지만 움직임이 많은 동적 블록의 경우 첫 번째 탐색이 잘못되었을 경우 국부적최적(local optimal)에 빠질 수 있는 단점을 가지고 있다. 다음 그림 2.5는 3단계 탐색 알고리즘을 이용하여 움직임 벡터를 탐색하는 경로를 나타내었다.

3단계 탐색 기법의 탐색 알고리즘은 다음과 같은 단계로 구성된다.

1단계: 탐색 영여의 중심점으로부터 4화소 간격의 9개의 탐색 후보 점 즉 현재 프레임의 매크로 블록과 이전 프레임의 탐색영역에서 8방향으로 4화소 간격에 위치한 9개의 탐색 후보 점에 대해 최소의 매칭 기준 에러값을 가지는 탐색 후보 점을 구한다.

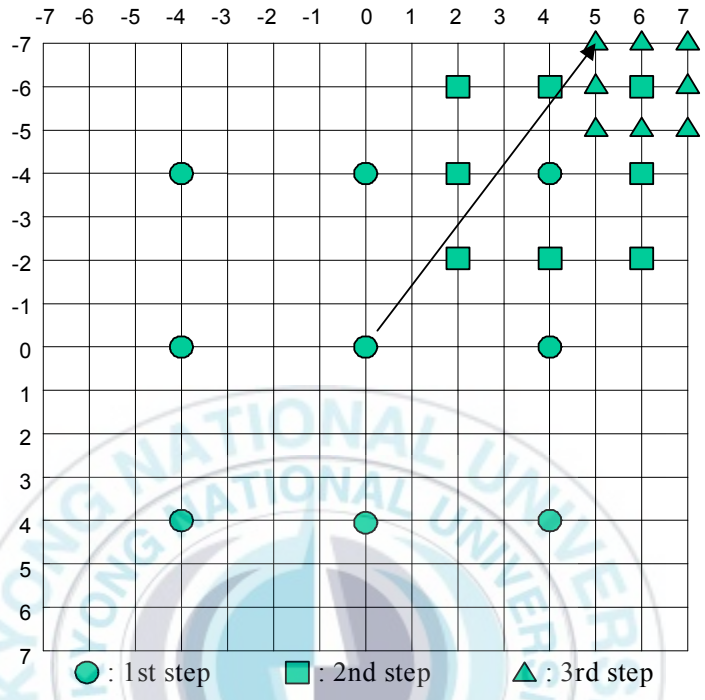

그림 2.5 3단계 탐색 방법

2단계: 1단계에서 구한 최소의 블록 매칭 오차값을 갖는 탐색 후보 점을 중심으로 탐색 후보 점과 간격을 이전 단계보다는 절반으로 줄어든 2화소 간격에 위치한 8개의 탐색 후보 점에 대해 최소의 블록 매칭 오차값을 갖는 탐색 후보 점을 구한다.

3단계: 2단계에서 구한 최소의 블록 매칭 오차값을 갖는 탐색 후보 점을 중심으로 탐색 후보 점과의 간격이 절반으로 줄어든 1화소 간격으로 8개의 탐색 후보 점에 대해 매칭오차를 계산하여 최소가 되는 탐색 후보 점을 구하여 최종적으로 해당 블록에 대한 움직임벡터로 결정한다.

2.3.2 4단계 탐색(four-step search) 방법

4단계 탐색 기법은 3단계보다 첫 번째 탐색 범위를 줄임으로써 국부적 최적에 빠지는 단점을 보완하고자 제안된 방식으로 2화소 간격의 5×5의 9개의 탐색 점을 가지는 사각형 형태로 분포하는 탐색 점들로 구성하며, 최종 탐색 블록의 크기를 15×15 탐색 영역의 중앙에 위치하는 2화소 간격의 5×5의 탐색패턴에 존재하는 그림 2.6에 1st step의 9개의 탐색 점을 초기 탐색 패턴으로 구성하고, 탐색패턴의 중심점이 최소 블록 매칭 오차값을 갖는 점으로 계산되면 4단계로 진행한다. 그렇지 않으면 2단계로 탐색을 진행한다.

2단계: 이전 단계의 최소 DM 점을 중심으로 탐색패턴을 2화소 간격의 5×5구조로 유지하면서 최소 블록 매칭 오차값을 계산한다. 최소 블록 매칭 오차값이 탐색 블록 중심점에서 발견되면 4단계로 진행하고 이전 최소 블록 매칭 오차값이 이전 탐색 패턴의 코너 점이라면 5개의 탐색점이 추가되고. 이전 최소 블록 매칭 오차값이 수직 또는 수평 방향이라면 3개의 탐색점이 추가된다.

3단계: 탐색 패턴은 2단계와 동일하지만 탐색 블록의 크기가 15×15로 고정되어 있으므로 무조건 4단계로 진행한다.

4단계: 탐색 블록의 코너 또는 모서리까지 탐색이 진행되면 탐색점간의 간격을 절반으로 줄어든 1화소 간격의 3×3의 탐색 패턴의 9개의 탐색 점에 대해 최소 매칭 기준 오차값을 갖는 탐색 점을 구하여 최종적으로 해당블록에 대한 동작벡터로 결정한다.

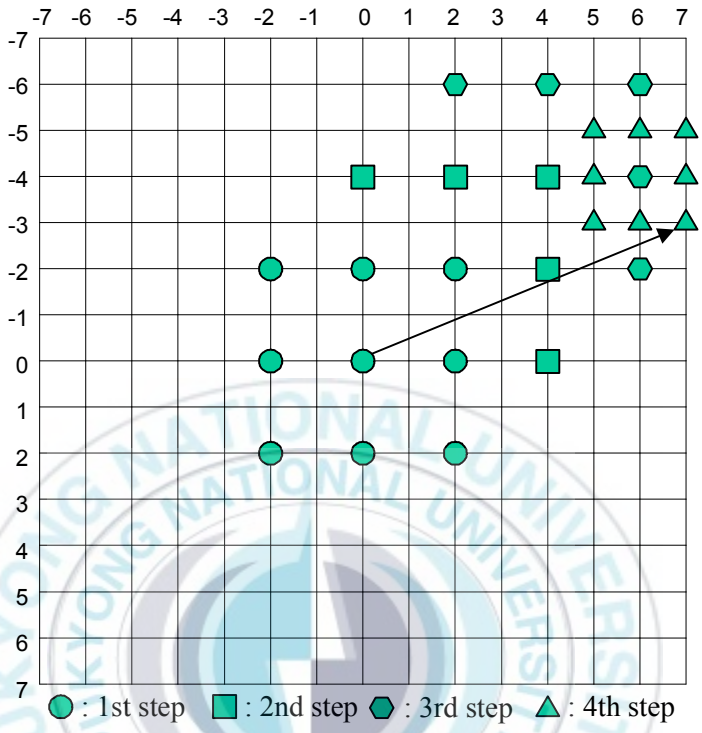

그림 2.6 4단계 탐색 방법

2.3.3 NTSS(new three step search) 방법

새로운 3단계 탐색 기법은 동작 벡터가 탐색영역의 중심에 분포한다는 사실을 이용하여 3단계 탐색 기법을 보완한 방식으로 정적블록과 동적 블록의 빠른 식별을 위해 중간-정지 기술을 이용하여 이들 블록의 동작을 추정하여, 중심점에 이웃하는 3×3크기의 8개 점을 탐색 후보 점으로 추가하여 총 17개의 탐색 후보 점에 대한 매칭 기준 오차값을 계산한다.

새로운 3단계 탐색 방법은 그림 2.7과 같이 나타낼 수 있다.

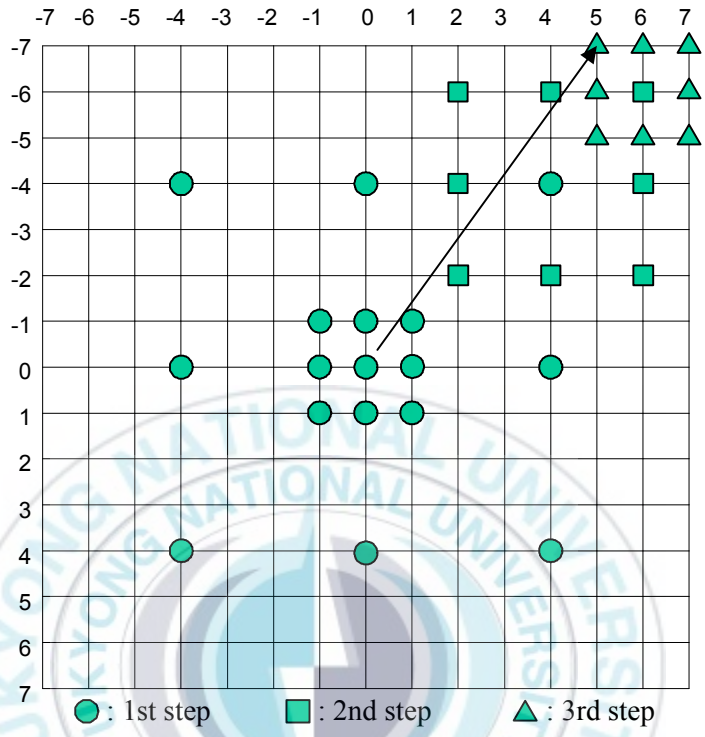

그림 2.7 NTSS 탐색 방법

새로운 3단계 탐색 알고리즘은 다음과 같이 구성된다.

1단계: 17개의 탐색 후보 점에 대해 최소 블록 매칭 오차값을 계산하여 최소 블록 매칭 오차값이 탐색 영역의 중심에서 발견되면 그 점을 동작벡터로 결정하고 탐색을 중지하며, 만일 최소 블록 매칭 오차값이 탐색영역의 중심점에 이웃하는 점이면, 최소 블록 매칭 오차값의 8개 이웃 점(8-neighbors)에 대해 추가적으로 매칭 기준 오차 값을 계산하고 최소 블록 매칭 오차값을 구하여 그 점을 동작벡터로 결정하고 탐색을 중지한다.

2단계: 1단계에서 구한 최소 블록 매칭 기준 오차값을 갖는 탐색 후보 점을 중심으로 사각형 형태의 탐색 영역에 대해 탐색을 실시하는데, 이때 탐색 후보

점들 사이의 간격을 이전 단계보다는 절반으로 줄어든 2화소 간격의 탐색 후보 점에 대해 매칭 기준 오차를 계산하여 최소값을 갖는 탐색 후보 점을 구한다.

3단계: 2단계에서 구한 최소 매칭 기준 오차값을 갖는 탐색 후보 점을 중심으로 탐색 후보 점들 간의 간격이 절반으로 줄어든 1화소 간격으로 8개의 탐색 후보 점에 대해 매칭 기준 오차를 계산하여 최소값을 갖는 점을 구하여 최종적으로 해당 블록에 대한 움직임 벡터로 결정한다.

2.3.4 2차원 로그형 탐색(two-dimensional logarithmic search) 방법

2차원 로그형 탐색(2D logarithmic search) 기법은 대부분의 움직임이 상하좌우로 일어난다는 사실을 이용한 탐색기법이다.

이 탐색 방법은 복잡한 움직임을 가진 영상보다는 수평방향이나 수직 방향으로 움직임이 일정하게 발생하는 영상의 탐색에 적합하다.

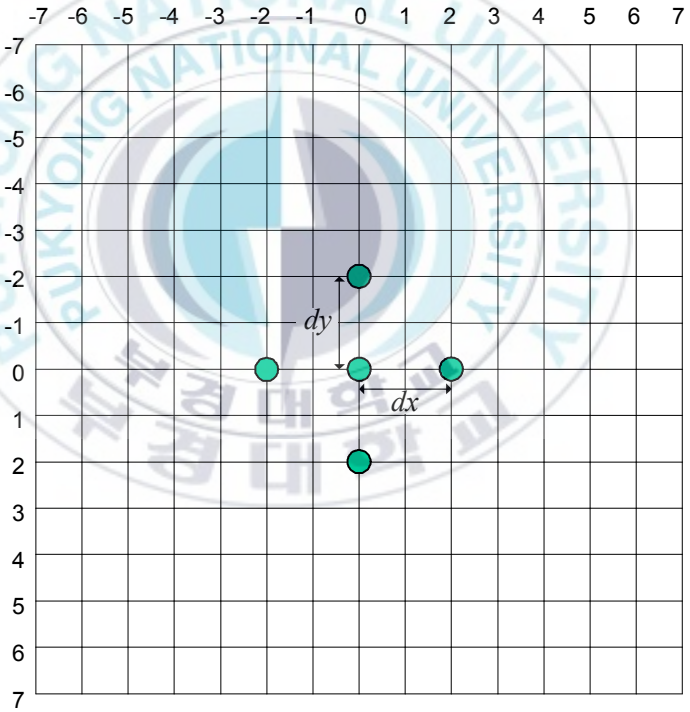

그림 2.8 2차원 로그형 탐색 방법

2.3.5 블록 기반 경사 하강 탐색(block-based gradient descent search) 방법

동영상 데이터의 특성상 동작을 포함하고 있는 블록이 탐색영역의 중심에 있을 확률이 높다는 가정 하에 탐색영역의 중심점을 포함하는 3×3화소로 이루어진 정사각형 블록을 블록 매칭의 시작으로 하여 최소값이 존재하는 영역으로 탐색을 전개하는 기법이다.

이 탐색 기법은 변이나 모서리 방향으로 꽉 채워진 탐색을 수행함으로 탐색의 속도는 빠르지 않으나 최적의 움직임 벡터를 찾을 확률이 높아서 탐색 속도보다 높은 화질을 요구할 때 적합한 방법이다.

블록기반경사 하강탐색 기법의 알고리즘은 다음과 같은 2단계의 간단한 구조로 구성된다.

1단계: 탐색영역의 중심점을 포함하는 3×3화소로 이루어진 정사각형 블록으로 구성된 탐색 패턴 내의 9개의 탐색 후보 점에 대해 매칭 오차를 계산한다.

2단계; 탐색 패턴 내에 존재하는 9개의 후보 점 중에서 탐색 블록의 중심점이 최소 블록 매칭 오차값이면 움직임 벡터로 결정되고 탐색을 종료한다. 이경우가 아니면, 최소 블록 매칭 오차값을 다음 탐색의 중심점으로 결정하고 1단계를 반복 수행한다.

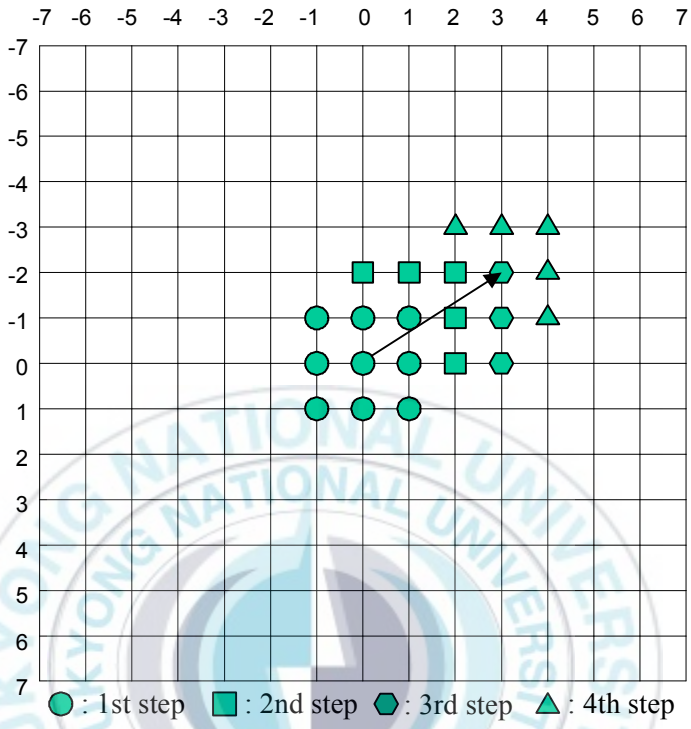

그림 2.9 BBGDS 탐색 방법

2.3.6 크로스 탐색 (cross search) 방법

화소 간격이 최대 동작변위 w의 반에 해당하는 사각형 형태의 대각점과 중심점으로 구성된 5개의 탐색 후보 점들로 탐색을 진행하며, 탐색 단계의 크기가 1이 될 때는 탐색 방향이 대각선 방향 또는 수평, 수직 방향이 될 수 있다. 이 탐색 기법의 특징은 임계치를 이용하여 탐색 대상 블록이 동작이 포함된 블록 인지 비동작 블록인지 우선적으로 분류하여 탐색의 효율성을 기할 수 있도록 설계되었다는 것이다.

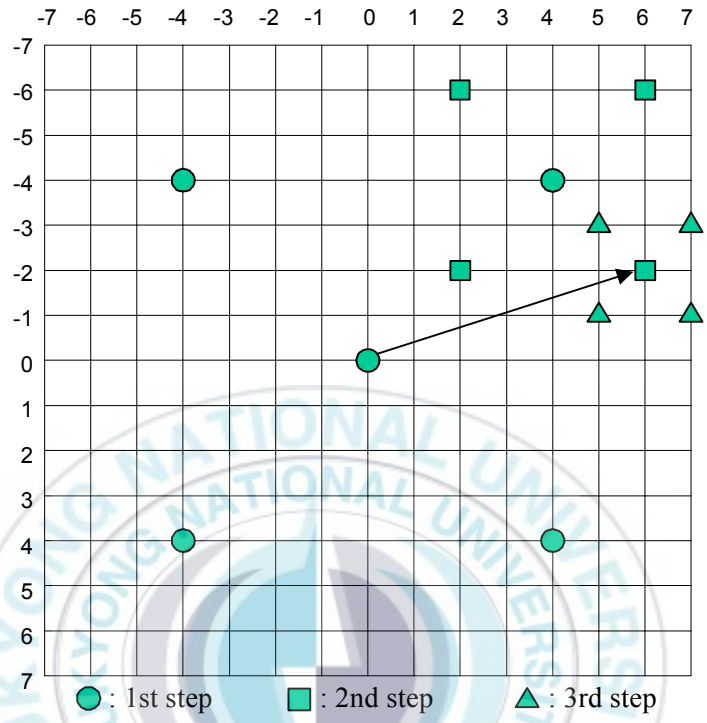

그림 2.10 크로스 탐색 방법

크로스 탐색 알고리즘은 다음과 같은 8단계로 구성된다.

1단계: 현재 탐색 블록과 $(0,0)$ 좌표를 포함하는 블록을 비교하여 블록 매칭 오차값의 값이 미리 정의된 임계치(threshold) T보다 작으면, 현재의 블록이 비동작 블록으로 분류되어 탐색을 중단한다. 아니면 실질적인 크로스 탐색 단계인 2단계로 진행된다.

2단계: 탐색 점 좌표 (x,y)를 $x=0, y=0$으로 초기화 하고 탐색 단계 크기(search step size) p를 최대 동작 변위 w의 반으로 설정한다.($p=w/2$)

3단계: 최소 블록 매칭 오차값 값을 가지는 위치 (x,y)를 탐색 중심점 좌표 (i,j)로 이동한다.($i=x, j=y$)

4단계: 5단계의 탐색 점 좌표 (i,j), $(i-p,j-p)$, $(i-p,j+p)$, $(i+p,j-p)$ $(i+p,j+p)$에서 최소값을 가지는 (x,y)를 찾는다.

5단계: 만약 $p=1$이면 6단계로 진행하고, 아니면 탐색 단계 크기 p를 최대 동작 변위 w의 반으로 설정하고($p=w/2$) 3단계로 돌아간다.

6단계: 이전 탐색 과정의 최종 탐색 점 (x,y)의 좌표가 (i,j), $(i-1,j-1)$, $(i+1,j+1)$ 중 하나이면 7단계로 진행하고 아니면 8단계로 진행한다.

7단계 좌표 (x,y), $(x-1,y)$, $(x,y-1)$, $(x+1,y)$, $(x,y+1)$에서 최소 블록 매칭 오차값을 가지는 점을 탐색한다. 이 좌표들은 St. Andrew's Cross라고 하며 이때 탐색된 최소 블록 매칭 오차값이 동작 벡터로 결정된다.

크로스 탐색 알고리즘을 이용한 탐색 경로는 그림 2.10과 같다.

2.3.7 다이아몬드 탐색(diamond search) 방법

다이아몬드 탐색 기법은 움직임벡터의 분포의 형태가 다이아몬드이며, 움직임 벡터가 일반적으로 탐색 영역의 중심으로부터 ±3 화소 이내에 포함된다는 특성을 이용함으로써 움직임 벡터를 찾을 확률을 높이려는 방식으로 움직임 벡터가 가운데 중심으로 분포되어 있는 영상의 경우에 적합하다. 그러나 급격한 움직임이 있는 영상의 경우에는 움직임 벡터가 ±3 화소 범위를 벗어나는 경우가 많기 때문에 움직임이 많은 영상의 경우에는 적합하지 않다.

이 탐색 알고리즘은 중심점으로부터 반지름 2화소를 가지는 원 내부에 포함된 13개 점 중에서 바깥 부분의 9개 탐색 점으로 그림 2.11(a)와 같은 큰 다이아몬드 형태의 탐색패턴을 구성한다. 수평 또는 수직 방향으로 탐색이 진행되면 5개의 점을 새로운 패턴에 추가하고 대각선 방향이면 3개의 점을 추가하여 탐색을 진행한 후 최종적으로 내부에 그림 2.11(b)와 같은 작은 다이아몬드

패턴을 구성하는 움직임 벡터를 결정하는 방법이다.

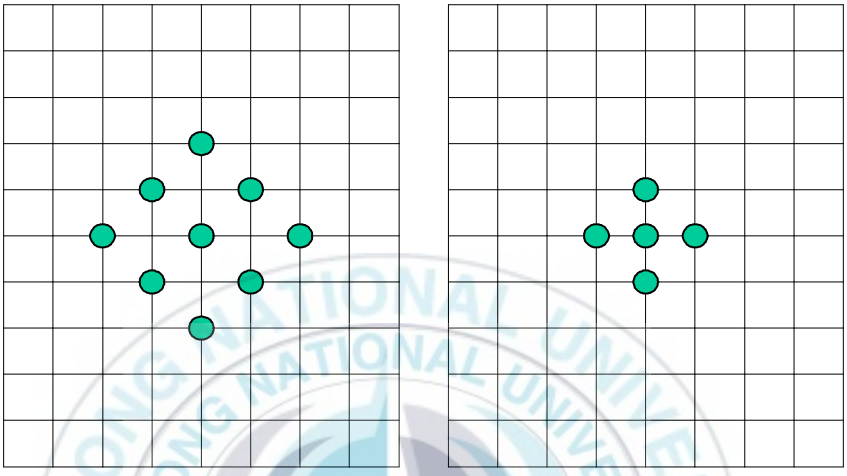

그림 2.11 다이아몬드 탐색 패턴
(a)큰 다이아몬드 탐색패턴 (b)작은 다이아몬드 탐색패턴

다이아몬드 패턴을 이용한 탐색 알고리즘은 다음과 같이 구성된다.

1단계: 탐색 영역의 중심점과 그 점을 중심으로 하는 다이아몬든 형태의 주위 9개의 탐색 점을 가지는 초기의 큰 다이아몬드 탐색 패턴(large diamond search pattern-LDSP)을 구성하여 최소 블록 매칭 오차값을 계산한다. 계산된 최소 블록 매칭 오차값이 중심점에 위치하면 3단계로 진행하고 그렇지 않으면 2단계로 진행한다.

2단계: 이전 단계에서 발견된 최소 블록 매칭 오차값을 새로운 LDSP의 중심점으로 지정하고 추가된 3개의 점 또는 5개의 점에 대하여 최소 블록 매칭 오차값을 계산한다. 새로 계산된 최소 블록 매칭 오차값이 패턴의 중심점에 위

치하면 3단계로 진행하고, 아니면 2단계를 반복한다.

3단계: 탐색 패턴을 이전 단계에서 탐색된 탐색 패턴의 중심점과 반지름 1 화소를 가지는 작은 다이아몬드 탐색 패턴(small diamond search pattern -SDSP)으로 변경하고 최소 블록 매칭 오차값을 계산한다. 이 단계에서 구한 최소 블록 매칭 오차값이 매칭블록이 나타나는 움직임 벡터가 된다.

다이아몬드 탐색 기법을 이용하여 동작벡터를 탐색해 나가는 과정의 예를 은 그림 2. 12의 그림과 같다.

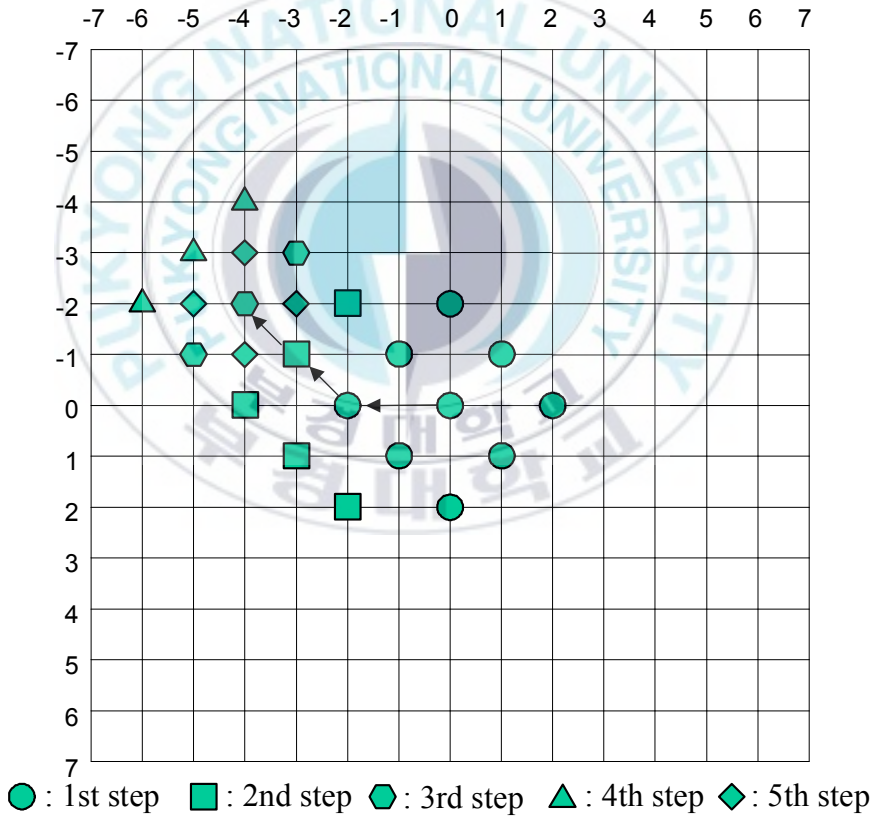

그림 2.12 다이아몬드 탐색 방법

2.3.8 크로스-다이아몬드 탐색(cross diamond search)

움직임 벡터가 일반적으로 탐색 영역의 중심점으로부터 ±3 화소 이내에 포함되어 있으며 그중에서도 중심좌표와 수평, 수직 좌표에 분포할 확률이 높은 것을 고려하여 빠른 탐색을 위해 제안된 방법으로 움직임 벡터가 탐색영역의 중심에 많이 분포 되어 있는 영상의 경우에 적합하다. 초기에 중심좌표(0,0)과 수직, 수평 방향의 2점씩, 즉 9개의 점으로 크로스 패턴을 적용하여 탐색 방향을 결정한 후 기존의 다이아몬드 패턴을 이용한 탐색을 진행하여 움직임 벡터를 구하는 방법이다.

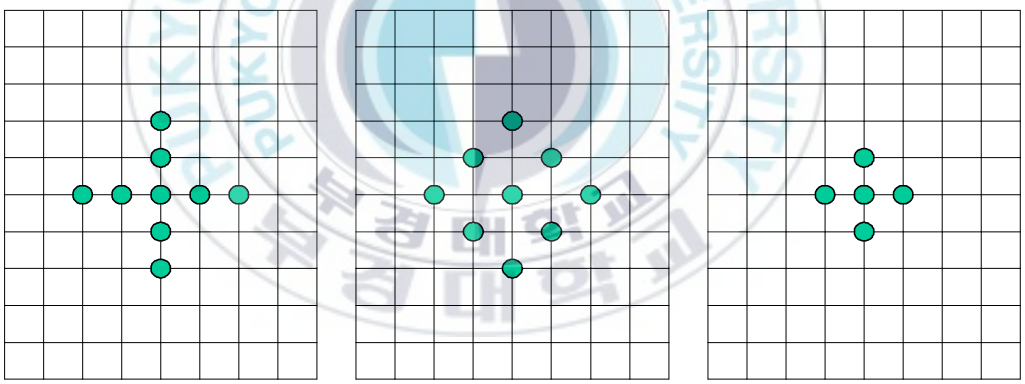

그림 2.13 크로스 다이아몬드 탐색 패턴
(a) 크로스 타입 (b) 큰 다이아몬드 타입 (c)작은 다이아몬드 타입

크로스-다이아몬드 패턴의 탐색 알고리즘은 다음과 같이 구성된다.

1단계: 탐색 영역의 중심점을 포함하는 크로스 패턴의 9개의 탐색 후보 점으로 최소 블록 매칭 오차값을 계산하여 탐색 패턴의 중심점이 최소 블록 매칭

오차값이면 탐색을 중단하고 중심점을 움직임벡터로 결정한다. 아니면 2단계로 진행한다.

2단계: 탐색 영역의 중심점을 중심으로 하는 큰 다이아몬드 탐색 패턴의 $(\pm 1, \pm 1)$에 위치하는 네 개의 탐색 후보 점 중 이전 단계에서 구해진 최소 블록 매칭 오차값에 인접한 두 개의 탐색 후보 점을 추가하여 최소 블록 매칭 오차값을 계산한다. 만약 최소 블록 매칭 오차값이 크로스 패턴의 중간날개(middle wing)에 위치하면 탐색을 중단하고 움직임 벡터로 결정한다. 그렇지 않으면 3단계로 진행한다.

3단계: 이전 단계에서 계산된 최소 블록 매칭 오차값을 탐색패턴의 중심점으로 설정하고 다이아몬드 형태의 주위 8개의 탐색 후보 점을 포함하는 LDSP를 구성하고 최소 블록 매칭 오차값을 계산한다. 계산된 최소 블록 매칭 오차값이 탐색패턴의 중심점에 위치하면 최종 5단계로 진행하고 아니면 4단계로 진행한다.

4단계: 이전 단계에서 발견된 최소 블록 매칭 오차값이 중심점에 위치하면 5단계로 가고, 아니면 4단계를 반복한다.

5단계: 탐색 패턴을 반지름이 1화소를 가지는 SDSP로 변경하고, 이 단계에서 구한 최소 블록 매칭 오차값점이 최소 블록 매칭 에러를 가지는 움직임벡터가 된다.

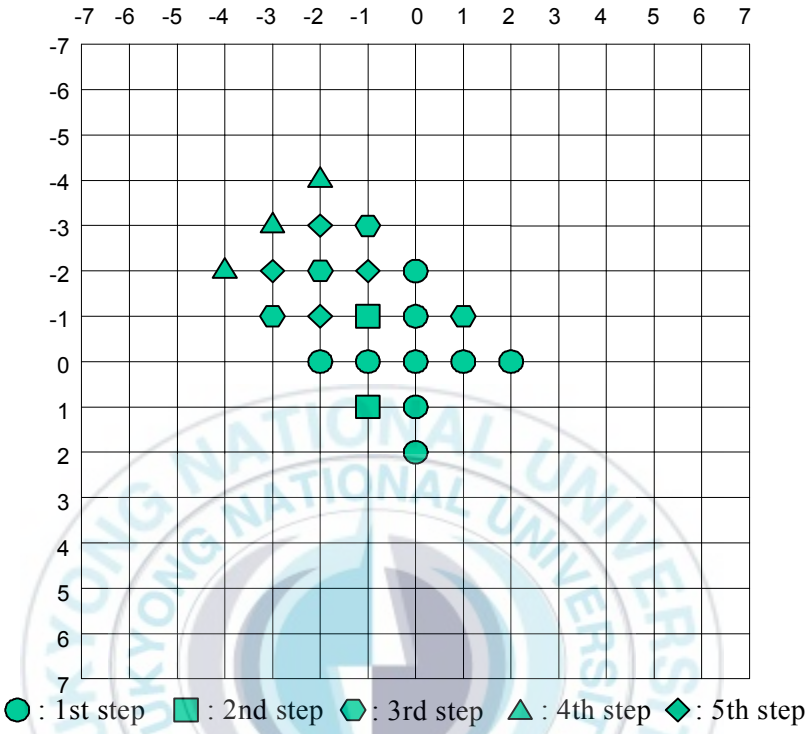

그림 2.14 크로스 다이아몬드 탐색 방법

2.3.9 육각형 기반 탐색(hexagon-based search) 방법

중심점과 수평의 거리가 2인 두 점과 4개의 중심점으로부터 거리 $\sqrt{5}$인 점으로 이루어진 그림 2.15(a)와 같은 큰 육각 탐색 패턴(large hexagon-based search pattern-LHEXBSP)과 중심점으로부터 거리 1인 4개의 탐색 후보 점을 갖는 그림 2.15(b)와 같은 작은 육각 탐색패턴(small hexagon-based search pattern-SHEXBSP)으로 움직임 벡터를 구하는 방법이다.

이 탐색 방법은 육각 패턴의 빠른 탐색 전개를 이용하는 방법으로서 비교적 움직임 벡터가 큰 영상에 적합하며 정적인 영상인 경우 빠른 전개를 통해

최적의 움직임 벡터가 아닌 움직임 벡터를 선택함으로써 화질의 저하를 야기할 수 있다.

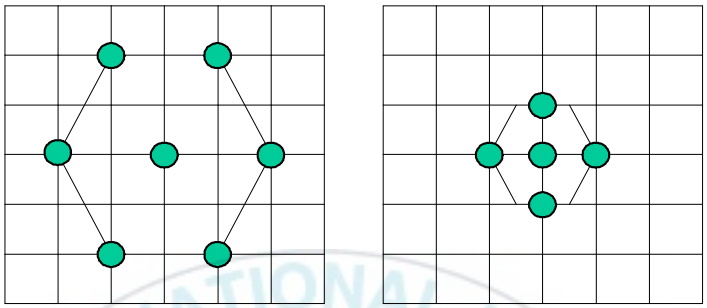

그림 2.15 육각형 기반 탐색 패턴
(a)큰 육각탐색 패턴 (b) 작은 육각 탐색 패턴

육각 패턴 기반 탐색 기법의 탐색 알고리즘은 다음과 같이 구성된다.

1단계: 탐색 영역에서 미리 정의 된 탐색 블록의 중심점 (0,0)을 중심점으로 하는 7개의 탐색 점을 가진 LHEXBSP가 주어진다. 최소 블록 매칭 오차값이 LHEXBSP의 중심점에 존재하면 3단계를 진행한 후 탐색을 종료하고, 아니면 2단계로 진행한다.

2단계: 이전 탐색 단계에서 발견된 블록 매칭 오차값을 새로운 LHEXBSP의 중심점으로 지정하여 3개의 새로운 후보점이 탐색되어 블록 매칭 오차값이 식별된다. 블록 매칭 오차값이 새롭게 구성되는 LHEXBSP의 중심점에 위치하면 3단계로 진행하고, 아니면 2단계를 반복한다.

3단계: 탐색패턴을 LHEXBSP에서 SHEXBSP로 변경한다. SHEXBSP에 포함된 4개의 점을 현재의 블록 매칭 오차값과 비교하여 새로운 블록 매칭 오차값이 동작 벡터로 결정된다.

탐색 알고리즘은 다음과 같이 나타낼 수 있다.

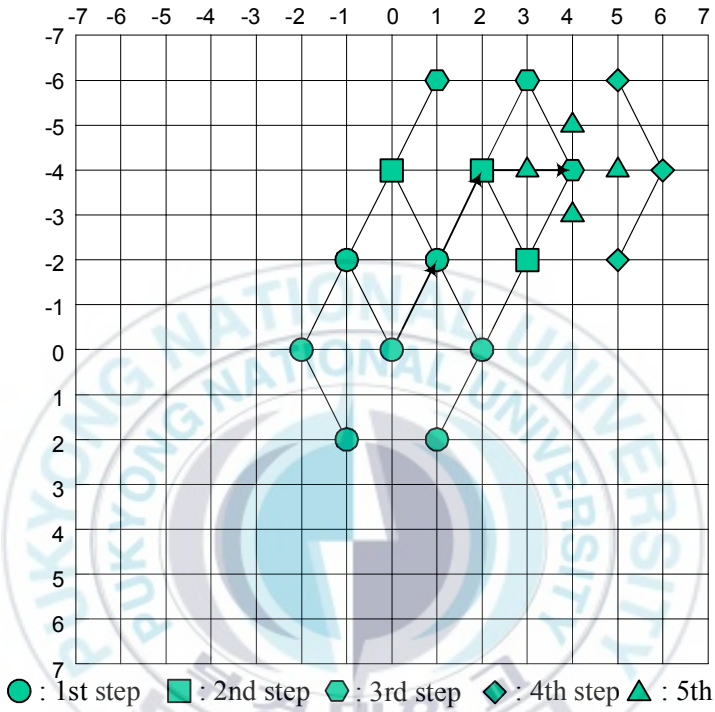

그림 2.16 육각형 기반 탐색 방법

2.4 고속 무손실 블록 매칭 방법

본 절에서 기술되는 알고리즘들은 전역 탐색의 계산량을 감소시키는 방법으로서 전역 탐색법과 똑같은 움직임 벡터를 찾으면서 계산량을 크게 감소시키는 알고리즘들이다.

2.4.1 SEA(successive elimination algorithm) 방법

SEA는 최적의 움직임 벡터를 찾기 위해 기본적으로 만족해야 하는 부등식과 그 부등식에 의해서 최적의 움직임 벡터가 될 수 있는 필요조건을 정의하고 그 조건을 만족하지 않는 탐색 점을 제외시킴으로써 계산량을 줄이는 방법으로서 매칭 기준으로 SAD를 사용하고 민코스키(Minkowski)의 부등식에 대하여 보다 좁은 경계 값을 적용함으로써 SAD계산으로부터 많은 영역을 제외시킨다. 이같이 좁은 경계를 구하는데 필요한 정보는 실질적인 움직임 추정에 앞서 효율적으로 계산되어진다. 수학적인 부등식의 유도과정은 다음절에서 상세히 기술한다.

(1) 알고리즘 개념

블록의 크기가 $N \times N$이고 탐색영역을 $(2N+1) \times (2N+1)$로 가정했을 때 최적의 움직임 벡터를 얻기 위하여 탐색영역 내에서 탐색과정을 제한하는 부등식을 세웠다. $f_t(i,j)$가 t번째 영상의 위치(i,j)에서의 휘도 값이고 $x,y(-N \le x,y \le N)$가 움직임 벡터이면 수학적인 부등관계로부터 식(2-2)를 얻을 수 있다.

$$\|f_t(x,y)|-|f_{t-1}(i-x,j-y)\| \le |f_t(x,y)-f_{t-1}(i-x,j-y)| \tag{2-2}$$

식(2-2)는 아래의 두 부등식과 같이 나타낼 수 있다.

$$\|f_t(x,y)|-|f_{t-1}(i-x,j-y)\| \le |f_t(x,y)-f_{t-1}(i-x,j-y)| \tag{2-3}$$

$$\|f_{t-1}(i-x,j-y)|-|f_t(i,j)\| \le |f_t(i,j)-f_{t-1}(i-x,j-y)| \tag{2-4}$$

한 블록 내의 모든 픽셀에 대하여 식(2-3)과 식(2-4) 양쪽의 합을 구하면 식(2-5), 식(2-6)과 같다.

$$\begin{aligned}\sum_{i=1}^{N}\sum_{j=1}^{N}|f_t(i,j)| - \sum_{i=1}^{N}\sum_{j=1}^{N}|f_{t-1}(i-x,j-y)| \\ \le \sum_{i=1}^{N}\sum_{j=1}^{N}|f_t(i,j)-f_{t-1}(i-x,j-y)|\end{aligned} \tag{2-5}$$

$$\begin{aligned}\sum_{i=1}^{N}\sum_{j=1}^{N}|f_{t-1}(i-x,j-y)| - \sum_{i=1}^{N}\sum_{j=1}^{N}|f_t(i,j)| \\ \le \sum_{i=1}^{N}\sum_{j=1}^{N}|f_t(i,j)-f_{t-1}(i-x,j-y)|\end{aligned} \tag{2-6}$$

식(2-5)에서 첫 번째 합은 참조 블록의 평균성분을 두 번째는 움직임 벡터 (x,y)의 SAD를 나타낸다. 그러므로 식(2-5)와 식(2-6)을 각각 식(2-7)과 식(2-8)로 표현할 수 있다.

$$R - C(x,y) \leq SAD(x,y) \tag{2-7}$$

$$C(x,y) - R \leq SAD(x,y) \tag{2-8}$$

현재까지 고려한 탐색 점 중 최소의 오차를 가지는 탐색 점, 즉 움직임 벡터를 $SAD_{\min}$을 알고 있다고 가정하자. 여기서 $SAD_{\min}$보다 $SAD(x,y)$가 더 작은 값을 가질 때 더 나은 움직임 벡터를 얻을 수 있다.

$$SAD(x,y) \leq SAD_{\min} \tag{2-9}$$

여기서, 식(2-9)를 식(2-7)과 식(2-8)에 대응시키면 다음과 같다.

$$R - C(x,y) \leq SAD_{\min} \tag{2-10}$$

$$C(x,y) - R \leq SAD_{\min} \tag{2-11}$$

위 수식은 다음과 같이 정리할 수 있다.

$$R - SAD_{\min} \leq C(x,y) \leq R + SAD_{\min} \tag{2-12}$$

식(2-12)는 SEA에 사용되는 중요한 결과이다. 이 수식은 (x,y)가 $R - SAD_{\min}$과 $R + SAD_{\min}$ 사이에 존재할 때 최적의 움직임 벡터가 될 수 있음

을 의미한다. 따라서 이 범위를 벗어나는 참조 블록은 움직임 추정에서 제외시킬 수 있다. 결국 SEA는 최적의 움직임 벡터를 배제하지 않고 탐색 과정의 계산적인 복잡도를 크게 줄일 수 있다. $C(x,y)$는 탐색 영역 내에 있는 $N\times N$블록의 평균값으로서 전체 계산량에 비해 많은 계산량을 필요로 하지 않는다. 이 알고리즘의 효율은 각 블록에 대한 평균 성분의 빠른 계산과 효과적인 초기 움직임 추정에 달려 있다.

(2) sum norm의 빠른 계산 기법

식(2-12)에 나타난 $C(x,y)$의 계산은 다른 움직임 추정 방법과 비교할 때, 하나의 오버헤드(overhead)에 속한다. 이 오버헤드의 계산량을 줄이기 위한 방법을 기술한다.

그림 2.17에 나타난 바와 같이 수평으로 한 화소만큼 이동된 블록에 대한 norm은 수평 이동전의 블록의 norm에 수평 이동에 의해 포함되는 N개의 화소값을 뺌으로써 간단히 구할 수 있다. 이러한 원리에 의해 norm의 계산에 많은 계산량이 소요되지 않는다. 이제 norm에 대한 계산을 전체 영상에 대해 생각해 보자. 우선 영상의 크기가 $I\times J, N\times N$블록 움직임 추정이 사용되고, unrestricted motion estimation을 고려하지 않는다고 가정해 보자. 그때 norm의 계산이 필요한 화소의 수는 총 $(I\times N)\times(J-N)$이다. 여기서 N만큼 뺄셈이 존재하는 것은 블록의 원점을 블록의 좌측상단으로 간주하므로 영상의 오른쪽과 아래쪽의 N개의 화소를 포함하는 블록의 원점이 $(I-N, J-N)$이 되기 때문이다. Norm의 계산이 $N\times N$의 mean filtering이 매우 유사함을 염두에 둔다면 계산량을 파악하는데 도움이 될 것이다.

1) 영상의 맨 위에 위치한 $I\times N$의 첫 번째 슬라이스(slice)에 대해 먼저 조사해 보자. 이슬라이스는 I개의 N차원 열에 대한 합을 $C11, C12, ..., C1I$라고 하

자. 이 계산에 요구되는 연산은 각 열은 $(N-1)$의 덧셈연산이 필요하고 이열이 I개 존재하므로 $I\times(N-1)$의 연산이 필요하다.

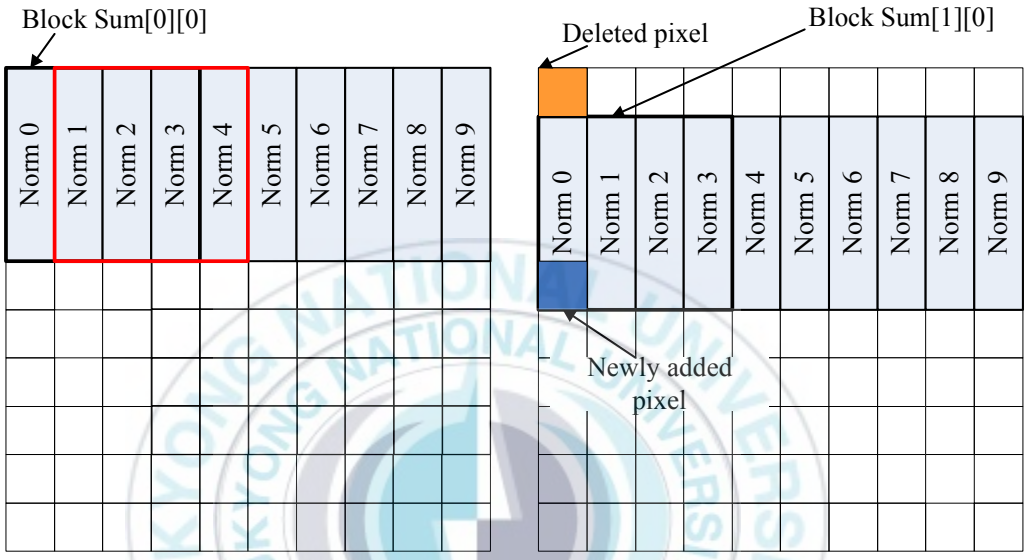

그림 2.17 Norm 계산

2) 두 번째 슬라이스에 대해 조사해 보면, 두 번째 슬라이스는 슬라이스가 수직방향으로 한 화소 이동된 $I\times N$ 크기의 블록을 의미한다. 즉, 첫 번째 슬라이스의 좌측상단 좌표가 (0,0)이라면, 두 번째 슬라이스의 좌표는 (0,1)이다. 두 번째 슬라이스에 포함된 블록들의 norm은 첫 번째 슬라이스의 norm을 이용함으로써 많은 계산량 없이 얻어질 수 있다. 즉, $C21 = C11 - f(1,0) + f(N+1,1)$. $C22 = C12 - f(1,2) + f(N+1,2)$, ..., $C2I = C1I - f(1,I) + f(N+1,I)$. 이 계산은 단순히 $2I$의 연산만을 필요로 한다. 이러한 계산을 두 번째 슬라이스 이후의 슬라이스들에 동일하게 적용이 가능하다. 그래서 열방향의 합을 구하는 데 필요한 총계산은 첫 번째 슬라이스에 필요한 계산량 $I\times(N-1)$과 첫 번째 슬라이스

를 제외한 $(J-N-1)$개의 슬라이스에 필요한 계산량 $2I\times(J-N-1)$의합이다. 즉 $I\times(N-1)+2I\times(J-N-1)$이다.

3)과정 1)과 과정 2)에서는 열 방향으로 합을 구하는데 필요한 계산량을 살펴보았다. 과정3에서는 열 방향으로 합을 이용하여 행방 향으로의 합을 구함으로써 $N\times N$ 블록의 합을 구하는데 필요한 계산량을 살펴보자. 우선 첫 번째 슬라이스에서 얻어진 열의 합 $C11, C12, ..., C1N$의합을 $SN11$이라고 하자. 그리고 수평방향으로 한 화소 이 동된 블록에 대한 합 $SN12$는 단순히 $C11$을 빼고 $C1(N+1)$을 더함으로써 얻을 수 있다. 즉 $SN12 = SN11 - C11 + C1(N+1)$. 이러한 계산은 $SN13$, ..., $SN1(I-N)$에 대해서도 적용 가능하다. 그래서 첫 번째 슬라이스에서 이 계산에 필요한 연산은 $(N-1)+2\times(I-N-1)$이다. 같은 방법으로 다음 슬라이스에 적용하면 총 $(J\times N)\times\{(N-1)+2\times(I-N-1)\}$의 연산을 필요로 한다.

따라서 영상 내에 있는 모든 $N\times N$블록의 norm을 계산하는데 필요한 계산량T는 다음과 같다.

$$\begin{aligned} T &= I(N-1)+2I(J-N-1)+(J-N)\{(N-1)+2(I-N-1)\} \\ &= 4IJ-(I-N)(N+3)-3I(N+1) \end{aligned} \tag{2-13}$$

영상내의 $N\times N$블록의 개수 B는 $(I/N)\times(J/N)$이므로 각 블록에 필요로 하는 계산량 G는 다음과 같다.

$$G = \frac{T}{B} = \left\{4 - \frac{(I-N)(N+3)}{IJ} - 3(N+1)\right\}N^2 \tag{2-14}$$

일반적으로 I와 J는 N에 비해 훨씬 크므로 G는 $4N^2$이다. 이 계산량은 $N \times N$블록들 간의 SAD계산에 필요한 계산량이 약 $2N^2$인 점을 감안한다면 2개의 탐색 점에 대한 SAD를 계산하는 정도의 오버헤드에 해당한다.

그림 2.18은 SEA알고리즘의 전체적인 흐름을 보여준다. 우선 전처리 과정으로 norm의 계산을 통한 후보블록의 제거를 통해 조건을 만족하지 못하는 후보블록을 우선적으로 제거한다. 그리고 조건을 만족하는 후보블록들과 최소 SAD값과의 비교를 통해 움직임 벡터를 보다 빠르게 구할 수 있다. 하지만 이 방법의 경우 SAD계산 이외에 전처리 과정에서 발생하는 부가적인 계산량이 발생하므로 아주 많은 계산량의 감소는 기대할 수 없다.

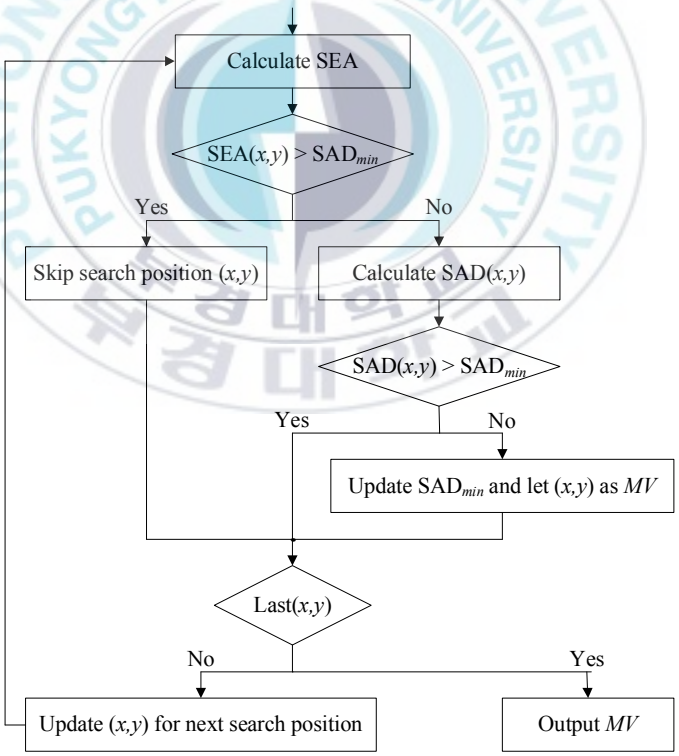

그림 2.18 SEA알고리즘 흐름도

2.4.2 MSEA(multilevel SEA) 방법

식(2-12)에 나타난 $C(x,y)$는 탐색영역 내에서 $N\times N$ 블록내의 화소 합으로서 이에 대한 부등식을 만족하지 않는 탐색 점을 제외시키는 방법이다. MSEA에서는 첫 번째 단계로서 $N\times N$블록에 대한 부등식으로부터 고려하지 않아도 되는 탐색 점을 제외시키고, 두 번째 단계로서 $N\times N$블록을 4개의 $N/2\times N/2$의 서브블록으로 나누고 서브블록의 화소 값의 합에 대한 부등식으로부터 고려하지 않아도 되는 탐색 점을 제외시킨다. 이러한 두 번째 단계의 과정을 반복함으로써 계산량을 크게 감소시키는 방법이 MSEA이다. MSEA의 가장 핵심이 되는 부등식을 이끌어 내기 위하여 다음과 같은 비용함수를 정의한다.

$$SSAD^k(x,y)=\sum_{i=1}^{2^k}\sum_{j=1}^{2^k}|f_t^k(i,j)-f_{t-1}^k(i+x,j+y)|\,k=0,1,2,...,P \qquad (2\text{-}15)$$

여기서

$$f_t^k(i,j)=f_t^k(2i-1,2j-1)+f_t^k(2i-1,2j)+f_t^k(2i,2j-1)+f_t^k(2i,2j) \qquad (2\text{-}16)$$

식(2-15)에서 $k=0$인 경우 블록의 합임을 알 수 있다. 즉 $SSAD^0(x,y)=R(x,y)$이다. Minkowski의 부등식 ($|A-B|\geq|A|-|B|$)을 적용하면 다음과 같은 부등식을 얻을 수 있다.

$$SSAD^0(x,y)\leq\cdots\leq SSAD^k(x,y)\leq\cdots\leq SSAD^{p-1}(x,y)\leq SAD^p(x,y) \qquad (2\text{-}17)$$

식(2-12)에서와 마찬가지로 현재까지의 탐색 위치 중 최선의 위치가 (m,n)이라고 하고 움직임 벡터 (m,n)의 추정 오차를 $SAD(m,n)$이라고 하자 이때 SAD를 $SSAD$ 로 표현하면 $SAD(m,n)=SSAD^{p}(m,n)$로 나타낸다. 만약 어떤 k에 $SSAD^{k}(m,n)$가 $SAD(m,n)$보다 크다면 (x,y)가 절대로 최적의 움직임 벡터가 될 수 없음을 의미한다. 역으로 말하면, 모든 k에 대해서 $SSAD^{k}(x,y)$가 $SAD(m,n)$보다 작다면 (x,y)가 최적의 움직임 벡터가 될 수 있음을 의미한다. 그래서 어떤 k에 대해 $SSAD^{k}(x,y)$가 $SAD(m,n)$보다 크다면 탐색 점에서 제외함으로써 계산량을 줄일 수 있다. 따라서 MSEA는 $k=0$에 대해 $SSAD^{0}(x,y)$가 $SAD(m,n)$ 보다 크다면 (x,y)를 탐색 점에서 제외시키고, 그렇지 않은 경우 $k=1$에 대해서 다시 $SSAD^{1}(x,y)$과 $SAD(m,n)$을 비교하여 (x,y)를 탐색 점에서 제외시킬지 결정하게 된다. 이러한 과정이 $k>1$에 대해서도 반복적으로 수행된다. $SSAD^{k}(x,y)$에 대한 부등식을 식(2-12)와 같이 표현하면 다음과 같다.

$$R- SAD_{\min} \le SSAD^{k}(x,y) \le R+ SAD_{\min} \tag{2-18}$$

모든 k에 대해 식(2-18)을 만족할 경우 (x,y)가 최적의 움직임 벡터 일 수 있으므로 탐색 점에서 제외시키지 않고, 모든 k에 대해 식(2-18)을 만족하므로 $SAD_{\min}$은 $SAD(x,y)$로 갱신되어 식(2-18)의 범위가 더욱 좁아진다.

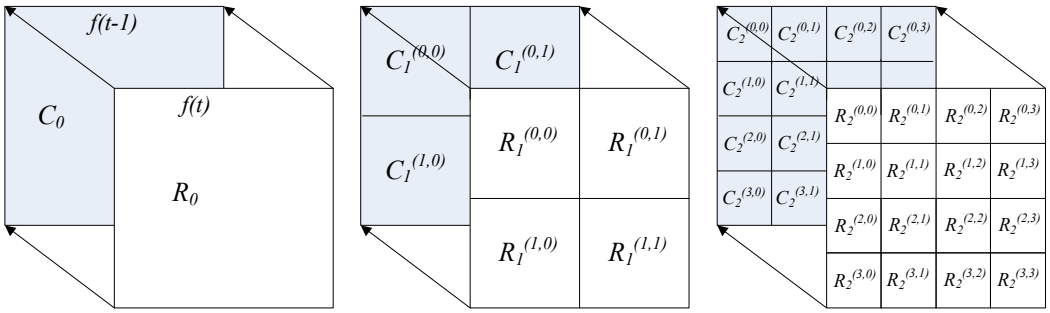

그림 2.19 각 레벨 별 MSEA

2.4.3 PDE(partial distortion elimination) 방법

이 방법은 하나의 블록의 완전한 계산을 하기 전에 부분 매칭 에러를 가지고 최소 에러와 비교함으로써 불필요한 계산을 줄이는 것이다. 즉, 매칭 에러의 중간 합이 그때까지의 최소 에러보다 크다면 나머지 계산을 할 필요가 없다는 것이다. 식(2-19)는 일반적인 SAD에서 각 행마다 중간 계산을 점검하도록 고친 것이다.

$$\sum_{i=1}^{k}\sum_{j=1}^{N}|f_t(i,j)-f_{t-1}(i+x,j+y)|\,k=1,\,2,\,\ldots,\,N \qquad (2\text{-}19)$$

앞에서 언급했듯이, k<N 일 때 부분 매칭 에러 합이 그때까지의 최소 에러보다 크다면 k+1 부터 N까지 행에 대한 매칭 에러 계산을 절약 할 수 있다. PDE 방법은 전역 탐색 방법뿐만 아니라 다른 고속 탐색 방법에서도 계산을 효

율적으로 줄이기 위해 사용되어질 수 있다. PDE 알고리즘의 계산의 감소는 주어진 탐색영역에서 전체 최소 에러를 가능한 빨리 찾는 것과 해당 매칭 블록에서 매칭 에러가 큰 영역을 먼저 계산함으로써 얻을 수 있다.

제 3 장 제안한 PDE기반 고속 움직임 추정 알고리즘

3.1 PDE와 나선형 스캔

대부분의 비디오 영상은 기본적으로 이전영상과의 움직임이 많지 않으므로 움직임 벡터 역시 움직임이 없는 경우가 많다. 그림 3.1은 동영상 데이터의 움직임 벡터가 나오는 것을 확률분포로 표시한 것이다. 그림에서 보면 대부분의 영상의 경우 움직임이 거의 없는 (0,0) 주위에 위치하는 경우가 50% 이상을 차지한다. 이러한 통계적인 특성을 보다 잘 이용하여 움직임 벡터를 효과적으로 추출하기 위해서 PDE의 변형된 알고리즘들은 시/공간적으로 인접한 움직임 벡터의 사용, 나선형 탐색방법, 그리고 이들 개념을 복합한 방법들을 사용한다.

그림 3.2는 움직임 벡터를 찾기 위해 필요한 나선형 스캔 방식을 보여준다. PDE알고리즘에서 중요한 것은 불가능한 후보 벡터들을 얼마나 빨리 검출해서 이를 제거 하느냐에 있다. 탐색 방법에 있어서는, 주어진 탐색 영역에서 나선형 탐색 방법을 사용한다. 나선형 탐색 방법을 갖는 PDE 알고리즘은 보다 빠르게 최소 SAD값에 도달 할 수 있고 최소SAD값과 나머지 후보 블록들의 SAD값의 비교를 통해 불필요한 후보블록을 보다 빠르게 제거할 수 있다. 그러므로 나선형 탐색 방법을 갖는 PDE알고리즘은 단순한 PDE 알고리즘보다 더 많은 계산량을 줄임을 알 수 있다. 따라서 제안하는 알고리즘에서는 나선형 탐색을 함께 사용 할 것이다.

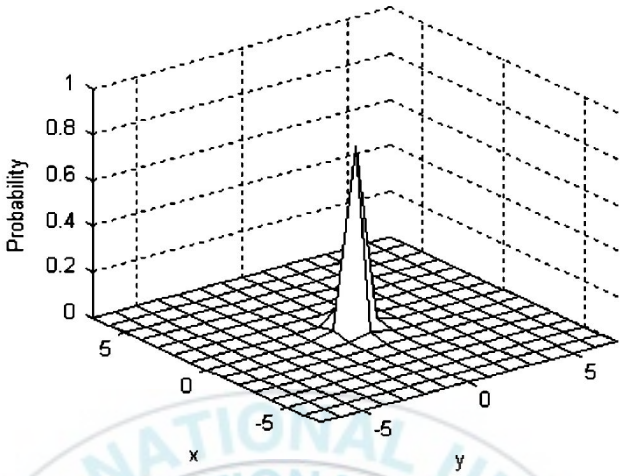

그림 3.1 움직임 벡터의 분포 확률

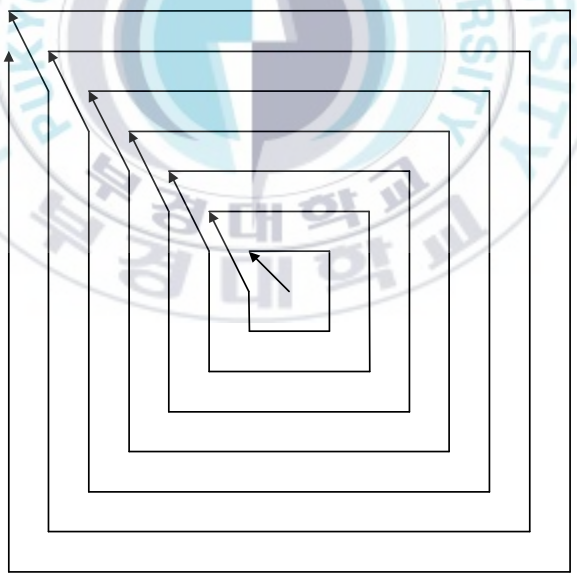

그림 3.2 나선형 스캔

3.2 최초부분계수 비교를 통한 적응적 고속 PDE알고리즘

우선 영상의 복잡도와 매칭에러의 관계를 정리하기 위해 Taylor급수 전개를 이용하여 매칭 에러와 기준 블록의 기울기 크기와의 관계를 정리 할 것이다. 가령, (t+1) 번째 프레임의 어떤 위치 $p=(x,y)$에서 화소값을 $\{f_{t+1}(p),p=(x,y)\}$이라 하자. 그리고 그 위치의 움직임벡터를 $mv=(mvx,mvy)$이라고 하자. 식(3-1)에서처럼 이전 프레임과 현재프레임 사이의 관계를 기술할 수 있다. 매칭 에러와 기준 블록의 기울기 크기간의 관계를 식(3-2)에서처럼 나타낼 수 있다. 여기서, $cmv=(cmvx,cmvy)$는 그 매칭 에러에 해당하는 후보 벡터를 나타낸다. 식(3-2)로 부터, 탐색 영역 안에서 기준 블록과 후보 블록의 매칭 에러는 기준 블록의 기울기 크기에 비례한다는 것을 알 수 있다. 즉, 영상의 복잡도에 따라 매칭 에러가 비례한다는 것을 의미한다.

$$f_{t+1}(p)=f_t(p+mv) \tag{3-1}$$

$$\begin{aligned} d_{t+1}(p) &= |f_{t+1}(p)-f_t(p+cmv)| \\ &= |f_t(p+mv)-f_t(p+cmv)| \\ &\approx \left|\frac{\partial f_t(p+mv)}{\partial x}(cmvx-mvx)+\frac{\partial f_t(p+mv)}{\partial y}(cmvy-mvy)\right| \\ &\approx \left|\frac{\partial f_{t+1}(p)}{\partial x}(cmvx-mvx)\right|+\left|\frac{\partial f_{t+1}(p)}{\partial y}(cmvy-mvy)\right| \end{aligned} \tag{3-2}$$

복잡도의 최적화를 위해 제안한 방법은 매칭 스캔에 영상의 복잡도를 잘

적용하기 위해 4×4서브 블록들을 이용한 방법이 1×16 벡터들을 이용한 방법보다 영상의 복잡도를 더 잘 영역화 할 것이다.

불필요한 후보블록을 미리 제거하기 위한 확률을 높이기 위해서는 효율적인 스캔방법을 찾아야 한다. 매칭에러가 큰 영역을 우선적으로 만날 확률을 높이기 위한 매칭 스캔 순서를 정하기 위해 최초의 SAD를 계산할 때 각 4×4서브블록의 부분 SAD를 계산하고 각 서브블록들 중 SAD값이 큰 순서대로 정렬하여 서브블록들 중에서 매칭에러가 큰 값이 우선적으로 나올 확률을 높여 값과 비교하여 고속으로 제거하는 알고리즘을 제안한다. 제안한 방법은 최초 후보블록의 SAD를 구할 때 부가적인 계산량 없이 4×4형태의 서브블록의 SAD값만을 가지고 이들을 비교하여 각 블록마다 적응적으로 매칭에러를 정렬함으로써 추가적인 계산량 없이 적응적인 매칭 스캔순서를 구할 수 있다. 기존의 변형된 PDE알고리즘에서 제시된 방법들은 4×4 서브 블록으로 나누어 순차적으로 스캔하는 방법, 후보블록간의 복잡도를 이용한 방법 그리고 4×4서브블록에 dithering order를 적용한 방법이 있다. 순차적인 방법은 매칭 에러를 구하는데 있어 매칭에러의 크기와 상관없이 순차적으로 비교하므로 후보블록간의 특성을 적용할 수 없는 단점이 있다. 후보블록간의 복잡도를 이용한 방법은 각 블록의 복잡도를 측정하기 위해 부가적인 계산을 필요로 한다. 검색할 후보점이 많을 경우 이 방법이 효율적이지만 최종 후보 점의 개수가 적을 때는 부적합하다. Dithering order를 이용한 방법 역시 후보블록간의 특징을 파악해서 적용할 수 없는 단점이 있다. 따라서 후보블록간의 특징을 파악하여 매칭에러가 큰 영역을 우선적으로 만날 확률을 높이는 스캔방법이 효율적이라 볼 수 있다. 이러한 매칭 스캔 순서를 정하는 방법으로 나선형 스캔을 적용하여 최초의 SAD를 계산할 때 4×4 형태의 서브블록의 매칭에러를 계산하고 각 서브블록들 중 에러

가 큰 순서대로 정렬하여 후보블록들 중에서 매칭에러가 큰 값이 우선적으로 나올 확률을 높여 고속으로 제거하는 알고리즘을 제안한다. 제안한 방법은 최초 후보블록의 SAD를 구할 때 부가적인 계산량 없이 4×4 형태의 서브블록의 SAD값만을 가지고 이들을 비교하여 각 블록마다 적응적으로 매칭에러를 정렬함으로써 추가적인 계산량 없이 적응적인 매칭 스캔순서를 구할 수 있다.

$$\sum_{s}^{k}\sum_{i=1}^{N/s}\sum_{j=1}^{N/s}|f_t(i,j)-f_{t-1}(i+x,j+y)| \quad (3\text{-}3)$$
$$k=1,\ 2,\ \ldots,\ N/s\times N/s,$$
$$s\in sorting\_order[\ ]$$

식(3-3)의 sorting_order는 SAD를 계산할 때 스캔순서를 형태의 서브 블록 별로 구해 큰 값을 기준으로 정렬하는 방법을 사용한다. sorting_order를 사용하여 정렬하여야 할 형태의 서브블록의 개수는 16개이다. 이들 16개의 후보블록들의 스캔순서를 각 블록별로 적응적으로 구하여 매칭에러가 큰 값이 우선적으로 나올 수 있는 확률을 높였다. 형태의 서브 블록의 부분SAD값들을 정렬하기 위해 사용되는 sorting_order에 사용되는 계산량은 무시할만한 아주 작은 부분이므로 무시하여도 좋을 정도이다. 제안한 방법과 같은 PDE 방법은 전역탐색 방법뿐만 아니라 다른 고속 탐색 방법에서도 계산을 효율적으로 줄이기 위해 사용되어질 수 있으므로 제안한 방법으로 SEA기반의 두 방법(SEA, MSEA)에 적용하여 계산량을 줄이고자 한다.

SEA에 제안한 방법을 적용하는 경우는 우선 SEA를 통하여 식(2-12)의 부등식의 경계조건을 사용하여 우선적으로 이 부등식을 만족하지 못하는 후보블록을 제거하고 경계조건을 만족하는 후보블록들 가운데 매칭에러가 큰 서브블

록이 나올 확률을 높이기 위해 각 블록별로 적응적 스캔 방법을 사용하여 보다 빠르게 후보 블록을 제거하였다.

MSEA의 경우 블록의 크기가 이고 서브블록의 크기가 이므로 식(2-18)에서처럼 연속해서 부등식의 레벨들을 적용해 나가면 서브블록의 크기보다 작아지게 된다. 따라서 제안한 방법은 Level 0과 Level 1까지 MSEA의 부등식의 경계조건을 만족하는 후보 블록들 가운데 매칭에러가 큰 서브블록이 나올 확률을 높이기 위해 각 블록별로 적응적 스캔 방법을 사용하여 보다 빠르게 후보 블록을 제거하였다.

또한 형태의 서브블록의 SAD값을 정렬하기위해 필요한 최초의 SAD값을 계산하기 위해영상의 시작일 경우에는 MV를 (0, 0)으로 지정하고 이후부터는 이전블록의 MV를 기준으로 SAD를 구하였다.

3.3 초기 임계치 비교를 통한 고속 PDE 알고리즘

기존의 변형된 PDE 알고리즘에서 제시된 방법들은 4×4 서브 블록으로 나누어 순차적으로 스캔하는 방법, 후보블록간의 복잡도를 이용한 방법, 그리고 4×4서브블록에 dithering order를 적용한 방법이 있다. 기존의 방법들은 후보블록간의 상관관계를 고려하지 않거나 복잡한 측정 등의 부가적인 계산이 필요한 단점이 있다. 따라서 후보블록간의 특징을 파악하여 매칭에러가 큰 영역을 우선적으로 만날 확률을 높이는 스캔방식이 효율적이라 볼 수 있다. 이러한 매칭 스캔 순서를 정하는 방법으로 나선형 스캔(spiral scan)을 적용하여 최초의 SAD를 계산할 때 4×4 형태의 서브블록의 매칭에러를 계산하고 각 서브블록들 중 에러가 큰 순서대로 정렬하여 후보블록들 중에서 매칭에러가 큰 값이 우선

적으로 나올 확률을 높여 고속으로 제거하는 알고리즘을 제안한다. 제안한 방법은 최초 후보블록의 SAD를 구할 때 부가적인 계산량 없이 4×4 형태의 서브블록의 SAD값만을 가지고 이들을 비교하여 각 블록마다 적응적으로 매칭에러를 정렬함으로써 추가적인 계산량 없이 적응적인 매칭 스캔순서를 구할 수 있다.

$$\sum_{s}^{k}\sum_{i=1}^{N/s}\sum_{j=1}^{N/s}|f_t(i,j)-f_{t-1}(i+x,j+y)| \quad (3\text{-}4)$$
$$k=1,\ 2,\ \ldots,\ N/s\times N/s$$
$$s\in matching\_order[\,]$$

식(3-4)의 *matching_order*는 SAD를 계산할 때 스캔순서를 최초 후보벡터의 서브 블록별 오차값의 크기순으로 정렬하는 방법을 사용한다. *matching_order*를 사용하여 정렬하여야 할 형태의 서브블록의 개수는 16개이다. 이들 16개의 후보블록들의 스캔순서를 각 블록별로 적응적으로 구하여 매칭에러가 큰 값이 우선적으로 나올 수 있는 확률을 높였다. 여기에 사용되는 계산량은 무시할만한 아주 작은 부분이므로 무시하여도 좋을 정도이다. 또한 불필요한 계산을 더욱더 많이 제거하기 위하여 최초의 부분 SAD계산 (SAD^1)과 최소 SAD값의 부분값을 추정하기 위하여 임계치(*Initial_Thr*)를 적용하여 임계치보다 크면 PDE계산을 중지하고 다음 후보블록을 계산하는 방법을 사용한다. 이 방법을 식으로 나타내면 식(3-5)와 같다. 여기에서 N은 서브블록의 총 개수이다(N=16).

제안한 알고리즘은 그림 3.3과 같은 순서로 진행된다. 우선 부분 SAD값을 계산하고 최소 부분SAD값과 임계치 값을 비교하여 불필요 계수를 우선적으로

제거한다.

$$SAD_1 = \sum_{i=1}^{1}\sum_{j=1}^{N}|f_t(i,j) - f_{t-1}(i+x, j+y)|,$$
$$SAD_1 \geq Initial\_Thr,\ \ Initial\_Thr = \frac{th}{N} SAD_{\min} \tag{3-5}$$

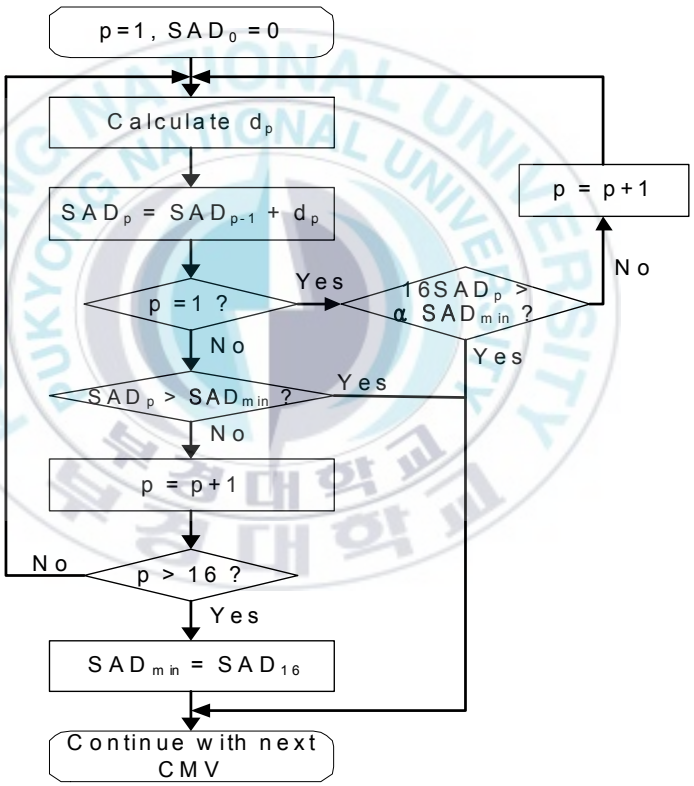

그림 3.3 초기 임계치 비교 방법의 흐름도

제안한 방법과 같은 PDE 방법은 전역 탐색 방법뿐만 아니라 다른 고속 탐

색 방법에서도 계산을 효율적으로 줄이기 위해 사용되어질 수 있으므로 제안한 방법을 MSEA에 적용하여 계산량을 줄이고자 한다. MSEA의 경우 블록의 크기가 16×16이고 서브블록의 크기가 4×4이므로 식(2-12)처럼 연속해서 부등식의 레벨들을 적용하기위해 Level 0과 Level 1까지 MSEA의 부등식의 경계조건을 적용하여 보다 빠르게 후보 블록을 제거하였다.

3.4 적응 부분 임계치 예측을 통한 고속 PDE 알고리즘

기존의 변형된 PDE 알고리즘에서 제시된 방법들은 후보 블록을 보다 효율적으로 빨리 제거하기 위하여 서브블록을 여러 가지 형태로 변형하거나 매칭 순서를 변형하여 보다 빨리 최소 SAD 값보다 큰 값이 나오도록 하였다. 서브블록의 형태는 MB을 크게 가로방향으로 분할하거나 또는 세로방향 그리고 4×4 형태의 정방 서브블록으로 나누고 서브블록의 각각의 픽셀들을 모아 부분계수로 만드는 방법 등이 있다. 제안한 방법은 부분 SAD값과 $SAD_{\min}$ 값을 비교하기 위하여 움직임이 발생한 영역에서의 오차값이 하나의 서브블록에 몰리지 않도록 normalized 방법을 사용하여 부분계수를 정렬한다. normalized 방법은 그림 3.4와 같이 16×16의 크기를 갖는 블록들에서 서브블록의 오차값을 영상 전체에 걸쳐서 구하게 된다. 서브블록별 SAD 값은 수식(3-6)과 같이 구할 수 있다. SAD값은 수식(3-7)과 같이 구할 수 있다.

$$d_p = \sum_{k=1}^{3}\sum_{l=1}^{3}\left| \begin{array}{l} f_t(i+4k+s_p, j+4l+t_p) \\ \quad - f_{t-1}(i+4k+s_p+x, j+4l+t_p+y) \end{array} \right| \tag{3-6}$$

$$SAD^p = \sum_{i=1}^{p} d_i \tag{3-7}$$

비디오 영상에서 이전프레임과 현재 프레임간의 상관관계를 실험적으로 고려해 보면 프레임간의 움직임은 매우 작다. 정적인 영상의 경우는 거의 움직임이 없는 경우도 있다. 이러한 비디오영상의 특성을 제대로 고려하기 위하여 매칭 스캔 순서를 정하는 방법으로 나선형 스캔(spiral scan)을 적용하여 최소 SAD값이 우선적으로 나올 수 있도록 고려하였다.

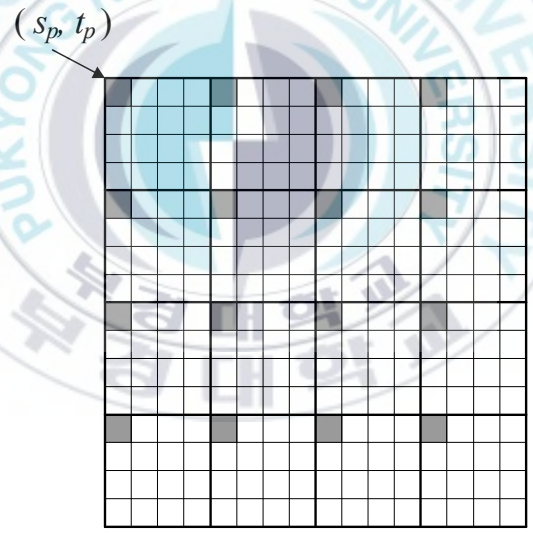

그림 3.4 16×16블록의 부분 오차 값 d_p의 계산순서

기존의 PDE방법은 부분 SAD값과 최소 SAD값의 비교를 통해 최소 SAD값보다 큰 값을 가지면 후보블록에서 제거하는 방법을 사용하였다. 제안한 방

법은 최초 SAD값을 통해 최소 SAD값의 예측을 각 블록별로 적응적으로 하여 불필요한 계산을 보다 효율적으로 줄였다. 최소 SAD값을 적응적으로 예측하기 위하여 우선 현재 블록과 인접 세 블록 간(왼쪽(left), 위쪽(top), 오른 위쪽(topright))의 초기 SAD값과 최소 SAD값의 비율을 통하여 현재 블록의 초기 SAD값으로 최소 SAD값 적응적으로 예측할 수 있다.

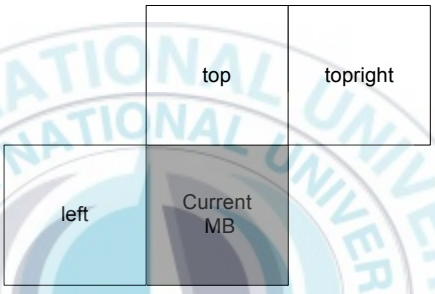

그림 3.5 인접블록간의 상관관계

식(3-8)에서 β는 인접 세 블록간의 초기 SAD와 최소 SAD의 관계를 나타낸다. γ는 현재 블록까지의 초기 SAD와 최소 SAD의 비율의 평균이다.

$$
\begin{aligned}
&SAD^{p} \geq AdpThr^{p}, \\
&AdpThr^{p} = \frac{1}{N} \times SAD_{\min} \times (\beta \times p + \gamma) \\
&\beta = \frac{SAD_{initial\_left} + SAD_{initial\_top} + SAD_{initial\_topright}}{SAD_{min\_left} + SAD_{min\_top} + SAD_{min\_topright}} \\
&\gamma = \frac{1}{n-1} \sum_{k=1}^{n-1} \frac{SAD_{initial}}{SAD_{\min}}
\end{aligned}
\tag{3-8}
$$

제안한 방법과 같은 PDE 방법은 전역 탐색 방법뿐만 아니라 다른 고속 탐색 방법에서도 계산을 효율적으로 줄이기 위해 사용되어질 수 있으므로 제안한 방법을 HEXBS에 적용하여 보다 빠르게 후보 블록을 제거하였다.

제 4 장 구현 및 결과

4.1 구현 방법

4장에서는 PDE기반의 제안한 방법을 가지고 여러 실험 영상에 적용해 본 결과의 구현 및 결과에 대해서 설명하고자 한다. 제안된 알고리즘과 기존의 알고리즘의 성능을 비교하기 위해 100프레임의 “foreman”, “car phone”, “trevor”, “akio”, “claire”, “grandmother”의 영상 시퀀스를 가지고 실험을 하였다. 매칭 블록의 크기는 16×16이며, 탐색 영역의 범위는 ±7 화소로 선택했다. 프레임의 크기는 QCIF(176×144)이다. 실험 결과는 계산된 평균 행의 수와 PSNR로 나타내었다. 계산된 평균 행의 수는 하나의 스캔 블록에서 계산 되는 평균 행의 수치이다. 표에서 나타난 수치는 복잡도를 측정하기 위해 모든 부가적인 계산을 고려한 값들이다. 여기서 테스트되는 모든 매칭 스캔 알고리즘은 탐색 영역에서 나선형 탐색 방법을 사용한 것이다. 이는 움직임 벡터들의 분포상태를 효율적으로 이용하는 방법으로서 PDE 및 SEA 알고리즘들에서는 널리 사용하고 있다.

4.2 최초부분계수 비교를 통한 적응적 고속 PDE알고리즘 적용

그림 4.1과 그림 4.2는 10 Hz의 “foreman” 과 “car phone” 영상 시퀀스에서 4×4 서브 블록에 대해 다양한 매칭 스캔의 계산된 평균 행의 수를 나타낸다. 이들 그림에서 주목할 것은 최초 부분계수 비교를 통한 적응 매칭방법이 종래의 순차적인 방법에 비해 매칭에러가 큰 서브블록이 우선적으로 나올 확률을 높여 불필요한 계산을 현저히 줄이는 것이다. 그리고 제안된 방법을 SEA기반 방법에 적용하였을 경우 보다 많은 계산량을 줄일 수 있다.

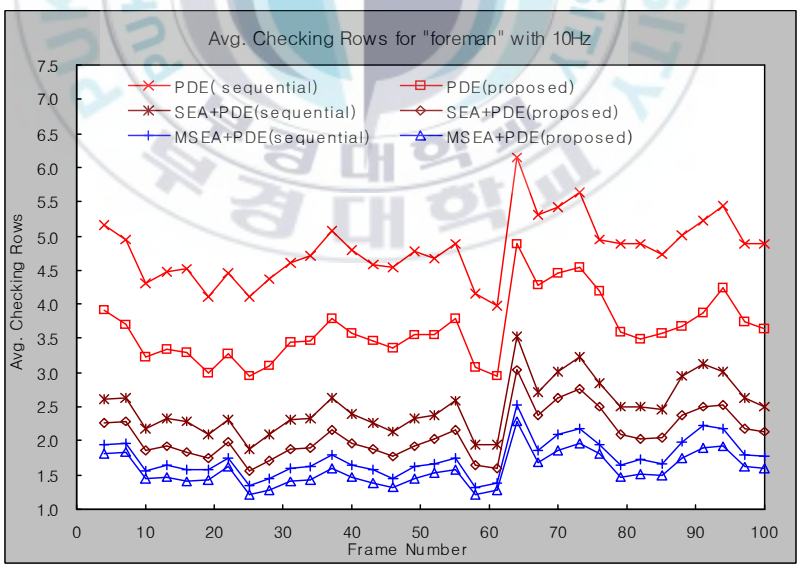

그림 4.1 10Hz에서 “foreman”영상에서의 최초 부분계수 비교방법을 적용한 평균 행의 수

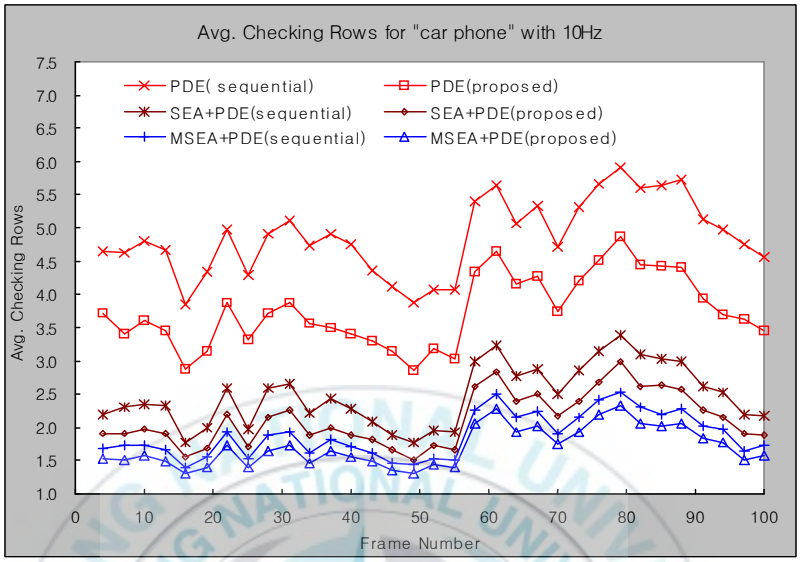

그림 4.2 10Hz에서 “car phone”영상에서의 최초 부분계수 비교방법을 적용한 평균 행의 수

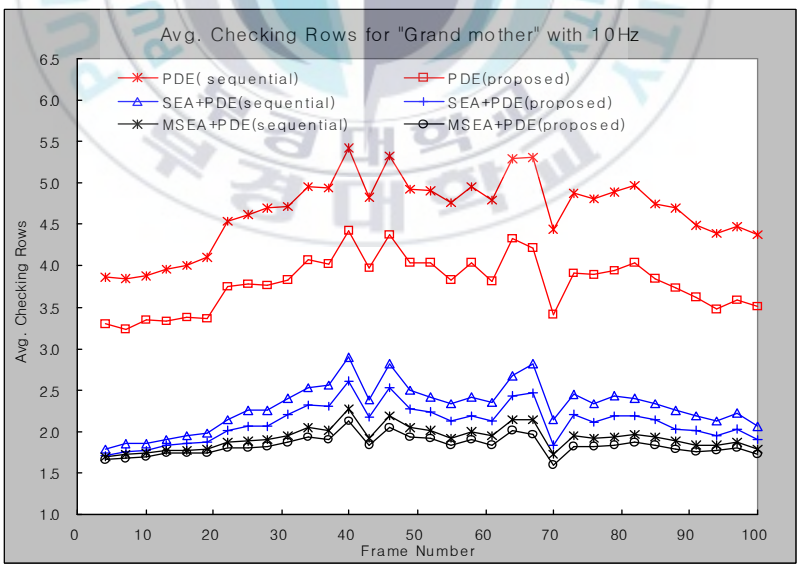

그림 4.3 10Hz에서 “grand mother” 영상에서의 최초 부분계수 비교방법을 적용한 평균 행의 수

표 4.1과 표 4.2는 30 Hz, 10 Hz의 모든 영상 시퀀스에서 서브 블록에 대해 다양한 방법에서의 순차적인 방법과 최초의 후보블록에서의 서브블록의 SAD값을 기준으로 한 적응적 매칭 스캔으로 계산된 평균 행의 수를 보여 준다. 제안한 방법은 기존의 전역탐색방법에 비해 78~92%의 계산량 감소를 보였고 SEA에 적용하였을 경우 88~96%, MSEA방법의 경우 89~96%의 계산량 감소를 보였다. 기존의 PDE방법과 비교하였을 경우 5~30%의 계산량 감축을 얻음을 볼 수 있었다.

표 4.1 30 Hz 영상 시퀀스에서 최초 부분계수 비교방법을 적용한 평균 행의 수

Algs. \ Seqs.	Foreman	Car phone	Trevor	Claire	Akio	Grand
Original FS	16.00	16.00	16.00	16.00	16.00	16.00
PDE(sequential)	4.13	3.91	3.21	4.04	1.72	3.65
PDE(proposed)	2.86	3.16	2.43	3.26	1.36	3.45
SEA+PDE(sequential)	2.48	2.51	1.77	1.57	0.78	2.39
SEA+PDE(proposed)	1.55	1.72	1.12	1.24	0.56	1.85
MSEA+PDE(sequential)	1.42	1.66	1.01	1.12	0.59	1.82
MSEA+PDE(proposed)	1.19	1.43	0.85	1.03	0.57	1.71

표 4.2 10 Hz 영상 시퀀스에서 최초 부분계수 비교방법을 적용한 평균 행의 수

Algs. \ Seqs.	Foreman	Car phone	Trevor	Claire	Akio	Grand
Original FS	16.00	16.00	16.00	16.00	16.00	16.00
PDE(sequential)	5.11	4.67	4.68	4.65	2.19	4.23
PDE(proposed)	3.64	3.75	3.53	3.54	1.54	3.80
SEA+PDE(sequential)	3.12	2.98	2.75	1.88	1.07	2.80
SEA+PDE(proposed)	2.10	2.12	1.84	1.44	0.69	2.10
MSEA+PDE(sequential)	1.88	1.99	1.60	1.29	0.71	1.99
MSEA+PDE(proposed)	1.58	1.71	1.32	1.16	0.63	1.83

4.3 초기 임계치 비교를 통한 고속 PDE 알고리즘 적용

그림 4.4, 그림 4.5는 10 Hz의 “foreman”, “car phone”영상 시퀀스에서 서브블록에 대해 다양한 방법으로 실험된 순차적 매칭 스캔과 최초의 후보 블록의 SAD값을 기준으로 한 적응적 매칭 스캔으로 계산된 평균 행의 수를 보여 준다. 제안한 방법을 적용한 경우 종래의 순차적인 방법에 비해 매칭 에러를 계산하는데 있어 매칭에러가 큰 서브블록이 우선적으로 나올 확률을 높여 임계치를 적용할 경우 불필요한 계산을 현저히 줄이는 것이다. 그리고 제안한 방법을 MSEA에 적용하였을 경우 불필요한 계산을 더 줄일 수 있었다.

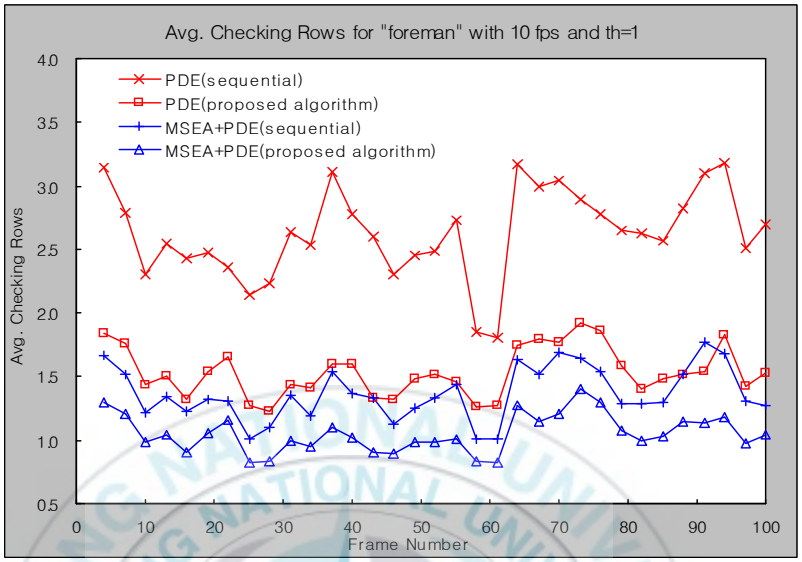

그림 4.4 10Hz th=1에서 “foreman” 영상에 초기임계치 방법을 적용한 경우 계산된 평균 행의 수

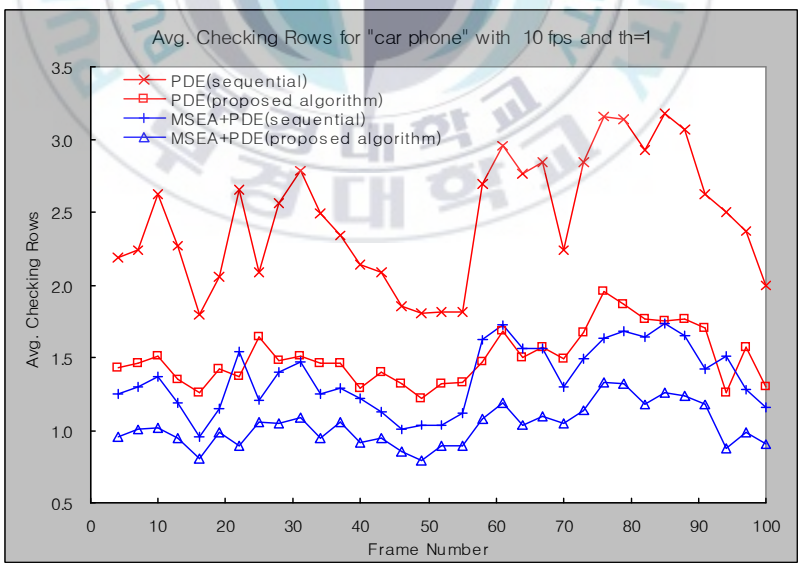

그림 4.5 10Hz th=1에서 “car phone” 영상에 초기임계치 방법을 적용한 경우 계산된 평균 행의 수

표 4.3~4.4는 30Hz에서 th=1, 4의 모든 영상 시퀀스에서 서브 블록에 대해 다양한 방법에서의 순차적인 방법과 제안한 방법을 적용하여 계산된 평균 행의 수를 보여 준다. 모든 수치는 복잡도를 측정하기위해 모든 부가적인 계산을 고려한 값이다. 고속 알고리즘을 전혀 사용하지 않은 원래의 전역 탐색 알고리즘은 표에서 보는 바와 같이 평균 계산 행의 수가 매칭 블록의 크기인 16이 된다.

표 4.3 10Hz, th=1일 때 영상 시퀀스에서 초기 임계치 방법을 적용한 경우 계산된 평균 행의 수

Seqs. / Algs.	Foreman	Car phone	Trevor	Claire	Akio	Grand
Original FS	16.00	16.00	16.00	16.00	16.00	16.00
PDE(sequential)	2.63	2.45	2.40	1.68	1.40	2.18
PDE(proposed)	1.53	1.50	1.43	1.15	1.14	1.40
MSEA+PDE(sequential)	1.37	1.36	1.16	0.79	0.70	1.28
MSEA+PDE(proposed)	1.05	1.03	0.89	0.69	0.63	0.98

표 4.4 10Hz, th=4일 때 영상 시퀀스에서 초기 임계치 방법을 적용한 경우 계산된 평균 행의 수

Seqs. / Algs.	Foreman	Car phone	Trevor	Claire	Akio	Grand
Original FS	16.00	16.00	16.00	16.00	16.00	16.00
PDE(sequential)	4.20	4.32	3.85	3.83	1.78	4.11
PDE(proposed)	2.87	3.04	2.72	2.80	1.29	3.15
MSEA+PDE(sequential)	1.80	1.94	1.48	1.28	0.75	1.99
MSEA+PDE(proposed)	1.59	1.70	1.30	1.21	0.69	1.87

표 4.3~4.4에서 제안한 알고리즘은 순차적인 매칭스캔 알고리즘에 PDE를 적용하였을 경우 87~92%의 계산량 감소를 보였고, MSEA방법의 경우 93~95%의 계산량 감소를 얻음을 볼 수 있었다. 표 4.5~4.6은 30Hz, th=1, 4에서의 전역 탐색 방법을 이용해서 얻은 예측영상을 원래의 영상과 비교해서 얻은 PSNR이다. PSNR의 차이는 th값의 차이에 의한 것이다. th=4의 경우 FS알고리즘과 비교하여 영상의 품질의 차이가 없었다. 따라서 제안한 방법은 부가적인 계산이 필요 없는 적응적 매칭 스캔 방법을 적용하여 화질의 감소 없이 계산량 감소를 이루었고 SEA기반 방법에 적용하였을 경우 보다 우수한 계산량의 감소를 얻을 수 있었다.

표 4.5 10Hz, th=1일 때 각 영상별 평균 PSNR

Algs. \ Seqs.	Foreman	Car phone	Trevor	Claire	Akio	Grand
Original FS	29.50	31.54	28.64	37.51	38.66	39.01
PDE(sequential)	29.24	31.18	28.25	37.23	38.55	38.84
PDE(proposed)	28.79	30.67	27.74	36.94	38.14	38.68
MSEA+PDE(sequential)	29.24	31.18	28.25	37.23	38.55	38.84
MSEA+PDE(proposed)	28.79	30.67	27.74	36.94	38.14	38.68

표 4.6 10Hz, th=4일 때 각 영상별 평균 PSNR

Algs. \ Seqs.	Foreman	Car phone	Trevor	Claire	Akio	Grand
Original FS	29.50	31.54	28.64	37.51	38.66	39.01
PDE(sequential)	29.50	31.54	28.64	37.51	38.66	39.01
PDE(proposed)	29.51	31.56	28.64	37.50	38.66	39.01
MSEA+PDE(sequential)	29.50	31.54	28.64	37.51	38.66	39.01
MSEA+PDE(proposed)	29.51	31.56	28.64	37.50	38.66	39.01

4.4 적응 부분 임계치 예측을 통한 고속 PDE 알고리즘 적용

그림 4.6~4.8은 10 Hz의 “trevor”, “car phone”, “clair”영상 시퀀스에서 서브 블록에 대해 다양한 방법으로 실험된 순차적 매칭 스캔과 부분 SAD 값에

적응적으로 임계치를 적용한 매칭 스캔으로 계산된 평균 행의 수를 보여 준다.

제안한 방법을 적용하면 각 후보 블록별로 종래의 순차적인 방법에 비해 매칭 에러를 계산하는데 있어 서브 블록 당 적응 임계치로 매칭에러 비교를 적용할 경우 불필요한 계산을 현저히 줄어들었다. 그리고 제안한 방법을 고속의 HEXBS에 적용하였을 경우 불필요한 계산을 더 줄일 수 있었다.

표 4.7~4.8은 30Hz, 10Hz에서 모든 영상 시퀀스의 서브 블록에 대해 순차적인 방법과 제안한 방법을 적용하여 계산된 평균 행의 수를 보여 준다. 모든 수치는 복잡도를 측정하기위해 모든 부가적인 계산을 고려한 값이다. 고속 알고리즘을 전혀 사용하지 않은 원래의 전역 탐색 알고리즘은 표에서 보는 바와 같이 평균 계산 행의 수가 매칭 블록의 크기인 16이 된다.

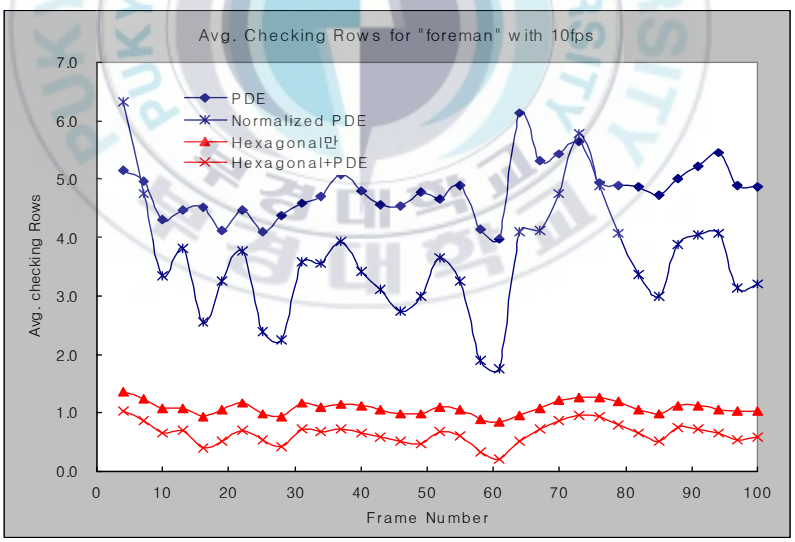

그림 4.6 10Hz에서 “foreman”영상에서의 적응 부분임계치 방법을 적용한 평균 행의 수

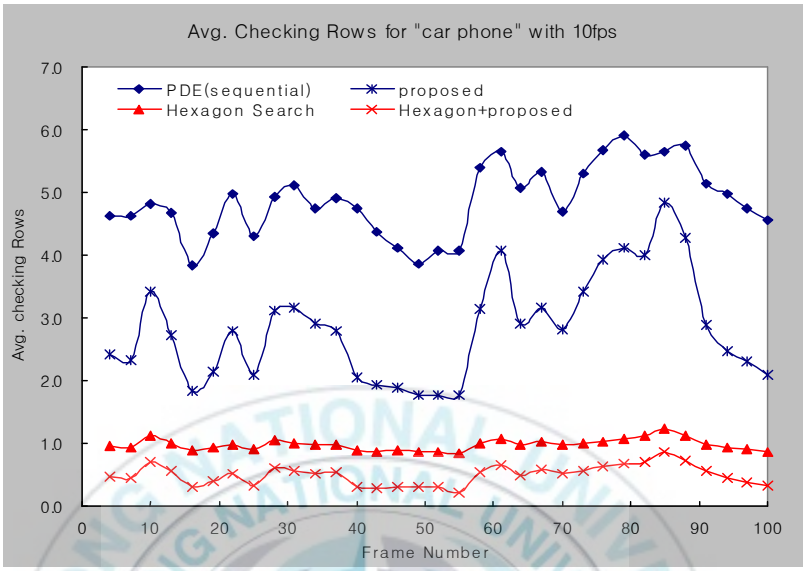

그림 4.7 10Hz에서 “car phone” 영상에서의 적응 부분임계치 방법을 적용한 평균 행의 수

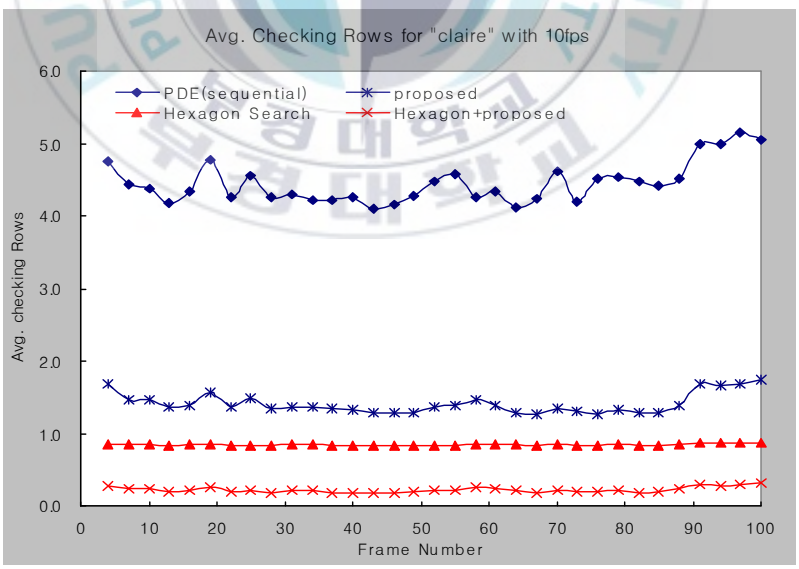

그림 4.8 10Hz에서 “clair” 영상에서의 적응 부분임계치 방법을 적용한 평균 행의 수

표 4.7~4.8에서 제안한 알고리즘은 기존의 PDE알고리즘보다 58~73%의 계산량 감소를 보였고, HEXBS방법에 적용할 경우 기존의 HEXBS방법보다 49~61%의 계산량 감소를 얻음을 볼 수 있었다. 표 4.8~4.9는 30Hz, 10Hz에서 기존의 방법과 제안한 방법에서의 PSNR이다. PSNR의 차이는 적응적으로 임계치를 구하였으나 발생하는 차이에 의한 것이다.

표 4.7 30Hz 영상 시퀀스에서 적응 부분임계치 방법을 적용한 평균 행의 수

Seqs. / Algs.	Foreman	Car phone	Trevor	Claire	Akio	Grand
Original FS	16.00	16.00	16.00	16.00	16.00	16.00
PDE(sequential)	4.04	4.24	3.18	3.94	1.71	4.18
PDE(proposed)	1.86	1.91	1.35	1.21	1.08	1.66
Hexagonal	0.90	0.89	0.85	0.83	0.83	0.85
Hexagonal+PDE(proposed)	0.35	0.32	0.21	0.17	0.14	0.22

표 4.8 10Hz 영상 시퀀스에서 적응 부분임계치 방법을 적용한 평균 행의 수

Seqs. / Algs.	Foreman	Car phone	Trevor	Claire	Akio	Grand
Original FS	16.00	16.00	16.00	16.00	16.00	16.00
PDE(sequential)	4.81	4.87	4.51	4.45	2.09	4.66
PDE(proposed)	3.60	2.83	2.35	1.41	1.15	1.86
Hexagonal	1.08	0.97	0.93	0.84	0.84	0.86
Hexagonal+PDE(proposed)	0.64	0.49	0.39	0.22	0.16	0.27

표 4.9 30Hz 영상 시퀀스에서 적응 부분임계치 방법을 적용한 PSNR

Algs. \ Seqs.	Foreman	Car phone	Trevor	Claire	Akio	Grand
Original FS	32.85	34.04	34.06	42.98	44.15	43.45
PDE(sequential)	32.85	34.04	34.06	42.98	44.15	43.45
PDE(proposed)	32.85	34.03	34.06	42.98	44.15	43.45
Hexagonal	32.12	33.63	33.89	42.94	44.11	43.42
Hexagonal+PDE(proposed)	32.12	33.61	33.88	42.94	44.11	43.42

표 4.10 10Hz 영상 시퀀스에서 적응 부분임계치 방법을 적용한 PSNR

Algs. \ Seqs.	Foreman	Car phone	Trevor	Claire	Akio	Grand
Original FS	29.50	31.54	28.64	37.51	38.66	39.01
PDE(sequential)	29.50	31.54	28.64	37.51	38.66	39.01
PDE(proposed)	29.50	31.54	28.63	37.51	38.66	39.01
Hexagonal	28.39	31.13	28.34	37.24	38.48	38.86
Hexagonal+PDE(proposed)	28.39	31.12	28.34	37.24	38.48	38.86

제 5 장 결 론

비디오 압축은 비디오 데이터의 저장 및 전송에 필수적인 역할을 한다. 비디오 압축의 기본 원리는 영상의 중복성을 제거 하는데 있다. 움직임 추정은 이러한 중복성 가운데 시간적인 중복성을 제거하는 필수적인 요소이다. 움직임 추정 가운데 전역 탐색 방식은 쉬운 하드웨어 구조와 최적의 성능을 가지지만 전역에 걸친 탐색 영역으로 인해 많은 계산량이 요구되고 실시간 비디오 전송에 문제가 된다. 이러한 문제점을 해결하기 위해 고속 움직임 추정 방법들이 제안되어 오고 있다. 고속 움직임추정 방법들은 전역 탐색 방법에 비해 예측화질의 손실을 감수 하더라도 계산량 감소에 중점을 두어 계산량 감소를 이루는 손실 움직임 추정 방법과 전역 탐색 방법에 비해 예측 화질의 손실 없이 계산량의 감소를 이루는 무손실 움직임 추정 방법이다. 손실 방법의 경우 지역 최소화 문제로 인한 정확하지 못한 움직임 벡터 때문에 예측 화질의 열화를 초래하고 특히 비트율이 제한된 응용분야에서 고속 알고리즘으로 인한 부정확한 움직임 벡터는 증가된 예측 에러로 인해 심각한 문제를 야기 시킬 수 있다. 최근의 손실 움직임 추정방법의 경우 계산량을 감소와 동시에 화질의 열화를 줄이는 쪽으로 연구가 진행되고 있다. 무손실 방법은 전역 탐색에 비해 예측화질의 손실이 발생하지 않으면서 보다 빠른 움직임 벡터를 찾을 수 있지만 후보 지점을 줄여서 움직임 벡터를 찾는 손실 방법에 비해 계산량 감소는 제한적이다. 보다 빠르게 후보블록을 제거하기 위해서는 초기 매칭 에러를 잘 선택하는 문제와 큰 부분 에러값을 갖는 서브블록을 우선적으로 비교할 수 있는 매칭방법에 관한 연구가 필요하다.

본 논문에서는 이러한 문제점들을 해결하기 위하여 PDE기반의 고속 움직임 추정 알고리즘들을 제안하였다. 기존의 전역 탐색 방법과의 비교하였을 때 예측 화질의 손실이 없이 보다 많은 계산량을 줄이는 방법과 손실이 발생하더라도 계산량을 보다 많이 줄이고 예측화질의 손실도 최소화 하는 방법들을 제안하였다. 우선 전역 탐색 방법과 비교하여 예측화질의 손실 없이 계산량을 줄이는 방법으로 첫 번째 방법은 기존의 PDE 방법의 계산량을 결정하는 서브블록의 매칭 순서를 각 MB별로 적응적으로 구하여 오차값이 큰 서브블록이 우선적으로 나오도록 하는 방법을 제안하였다. 제안한 방법을 PDE에 적용하였을 경우 FS방법과 비교하여 전역 탐색방법에 비해 78~92%의 계산량 감소를 보였고 SEA에 적용하였을 경우 88~96%, MSEA방법의 경우 89~96%의 계산량 감소를 보였다.

전역 탐색 방법과 비교하여 예측화질의 손실이 발생하더라도 계산량을 보다 많이 줄이고, 예측화질의 손실을 최소화 하는 방법으로 두 가지 방법을 제안하였다. 두 번째 방법은 서브 블록의 오차값 계산 시 초기 매칭 서브 블록과 최소 오차값에 임계치를 적용하여 불필요한 후보블록을 제거하는 고속 움직임 추정 방법을 제안하였다. 세 번째 방법은 서브 블록별로 최소 오차값의 적응적 예측을 통한 적응임계치를 각 서브블록별로 적용하는 고속 움직임 추정 방법을 제안하였다. 이 방법들은 보다 많은 계산량 감소를 이루었고 예측화질의 열화는 무시할 만한 수준이다. 두 번째 방법인 서브블록 초기 매칭 방법의 경우 전역 탐색 방법과 비교하여 87~92%, MSEA방법에 제안한 방법을 적용하였을 때 93~95%의 계산량 감소를 얻음을 볼 수 있었다. 세 번째 방법인 서브블록별 적응적 임계치를 적용의 경우 전역 탐색 방법에 비해 88~94%의 계산량 감소를 보였고, HEXBS방법에 제안한 방법을 적용할

경우 전역 탐색에 비해 96~99%의 계산량 감소를 얻음을 볼 수 있었다. 제안한 알고리즘들은 MPEG-2 및 MPEG-4 AVC를 이용하는 비디오 압축 응용분야에 유용하게 사용될 수 있을 것으로 사료된다.

참 고 문 헌

[1] H.G. Musmann, P. Pirsch, and H.J. Grallert, "Advances in picture coding," *Proc. IEEE,* vol. 73, pp. 523-548, Apr. 1985.

[2] R. Srinivasan and K. R. Rao, "Predictive coding based on efficient motion estimation," *IEEE Trans. Commun.*, vol. COM-33, no. 8, pp. 888-896, Aug. 1985.

[3] F. Dufaus and F. Moscheni, "Motion estimation techniques for digital TV: A review and a new contribution," *Proc. IEEE*, vol. 83, pp. 858-876, Jun. 1995.

[4] G. C. de Oliveira and A. Alcaim, "On fast motion compensation algorithms for video coding," *Proc. PCS*, pp. 467-472. 1997.

[5] J.Y. Lu, K.S. Wu, and J.C. Lin, "Fast full search in motion estimation by hierarchical use of Minkowski's inequality (HUMI)," *IEEE Trans. Pattern Recog.*, vol. 31, pp. 945-952, 1998.

[6] M.Z. Coban and R.M. Mersereau, "A fast exhaustive search algorithm for rate-constrained motion estimation," *IEEE Trans. Image Processing*, vol. 7, pp. 769-773, May 1998.

[7] S. Eckart and C. Fogg, "ISO/IEC MPEG-2 software video codec," *Proc. SPIE*, vol. 2419, pp. 100-118, 1995.

[8] ITU-T Recommendation H.263 software implementation, *Digital Video Coding Group at Telenor R&D,* 1995.

[9] A.M. Tekalp, *Digital Video Processing, Prentice Hall*, pp. 72-129, 1995.

[10] Y. Wang and J. Ostermann, "Comparison of block-based and mesh-based motion estimation algorithms," *Proc. IEEE ISCS*, pp. 1157-1160, 1997.

[11] C.H. Hsieh, P.C. Lu, J.S. Shyn, and E.H. Lu, "Motion estimation algorithm using inter block correlation," *IEE Electron. Letters,* vol. 26, pp. 276-277, Mar. 1990.

[12] L.W. Lee, J.F. Wang, J.Y. Lee, and J.D. Shii, "Dynamic search-window adjustment and interlaced search for block matching algorithm," *IEEE Trans. Circuits Syst. Video Technol.,* vol. 1, pp. 378-385, Feb. 1993.

[13] K. Sauer and B. Schwartz, "Efficient block motion estimation using integral projections," *IEEE Trans. Circuits Syst. Video Technol.,* vol. 6, pp. 513-518, Oct. 1996.

[14] X. Lu and O.C. Au, "An improved fast feature-base block motion estimation," *Proc. ICIP,* pp. 741-744, 1997.

[15] L.K. Liu and E. Feig, "A block-based gradient descent search algorithm for block motion estimation in video coding," *IEEE Trans. Circuits Syst. Video Technol.,* vol. 6, pp. 419-422, Aug. 1996.

[16] S.S. Lee and S.I. Chae, "Motion estimation algorithm exploiting low-resolution quantization," *IEE Electron. Letters,* vol. 32, pp. 647-648, Mar. 1996.

[17] Y.J. Baek, H.S. Oh, and H.K. Lee, "An efficient block matching

criterion for motion estimation and its VLSI implementation," *IEEE Trans. Consumer Electronics,* vol. 42, no. 4, pp. 885-892, Nov. 1996.

[18] B. Tao and M.T. Orchard, "Feature-accelerated block matching," *Proc. SPIE,* vol. 3309, pp. 469-476, 1997.

[19] L.J. Luo, C. Zou, and X.Q. Gao, "A new prediction search algorithm for block motion estimation in video coding," *IEEE Trans. Consumer Electronics,* vol. 43, no. 1, pp. 56-61, Feb. 1997.

[20] L. Luo, C. Zou, and Z. He, "A new prediction search algorithm of block motion estimation for video coding," *Proc. ICASSP,* pp. 1175-1178, 1997.

[21] S. Cucchi and D. Grechi, "A new features-based fast algorithm cor motion estimation: decimated integral projection(D.I.P)," *Proc. ICICS,* 1997.

[22] D.W. Kim, J.S. Choi, and J.T. Kim, "Adaptive motion estimation based on spatio-temporal correlation," *Signal Processing: Image Comm.* vo1. 13, no. 2, pp. 161-170, 1998.

[23] J.C. Tsai, C.H. Hsieh, S.K. Weng, and M.F. Lai, "Blockmatching motion estimation using correlation search algorithm," *Signal Processing: Image Comm.* vo1. 13, pp. 119-133, 1998.

[24] J.B. Xu, L.M. Po, and C.K. Cheung, "Adaptive motion tracking block matching algorithms for video coding," *IEEE Trans. Circuits Syst. Video Technol.,* vol. 9, no. 7, pp. 1025-1029, Oct. 1999.

[25] H.S. Wang and R.M. Mersereau, "Fast algorithms for the estimation of

motion vectors," *IEEE Trans. Image Processing*, vol. 8, pp. 435-438, Mar. 1999.

[26] B. Erol, F. Kossentini, and H. Alnuweiri, "Efficient coding and mapping algorithms for software-only real-time video coding at low bit rates," *IEEE Trans. Circuits Syst. for Video Technol.*, vol. 10, pp. 843-856, Sep. 2000.

[27] B. Liu and A. Zaccarin, "New fast algorithms for the estimation of block motion vectors," *IEEE Trans. Circuits Syst. Video Technol.,* vol. 3, pp. 148-157, Apr. 1991.

[28] L. W. Lee, J. F. Wang, J. Y. Lee, and J. D. Shie, "Dynamic search-window adjustment and interlaced search for block-matching algorithm," *IEEE Trans. Circuits Syst. Video Technol.,* vol. 3, no. 1, pp. 85-87, Feb. 1993.

[29] J. Feng, K.T. Lo, H. Mehrpour, and A.E. Karbowiak, "Adaptive block matching motion estimation algorithm for video coding," *IEEE Electron. Letters,* vol. 32, pp. 1542-1543, Aug. 1995.

[30] M.J. Chen, L.G. Chen, T.D. Chiueh, and Y.P, Lee, "A new block-matching criterion for motion estimation and its implementation," *IEEE Trans. Circuits Syst. Video Technol.,* vol. 5, pp. 231-236, Jun. 1995.

[31] H.S. Oh and H.K. Lee, "Adaptive adjustment of the search window for block-matching algorithm with variable block size," *IEEE Trans. Consumer Electronics,* vol. 44, no. 3, pp. 659-666, Aug. 1998.

[32] J.b. Xu, L.M. Po, and C.K. Cheung "A new prediction model search algorithm for fast block motion estimation," *Proc. KIP,* pp. 610-613, 1997.

[33] J. Lu and M. L. Liou, "A simple and efficient search algorithm for block-matching motion estimation," *IEEE Trans. Circuits Syst. Video Technol.,* vol. 7, pp. 429-433, Apr. 1997.

[34] S.R. Pan, S.S. Chae, and R.H. Park, "VLSI architectures for block matching algorithms using systolic arrays," *IEEE Trans. Circuits Syst. Video Technol.,* vol. 6, pp. 67-73, Feb. 1996.

[35] S.S. Lee and S.I. Chae, "New motion estimation algorithm and its block-matching criteria using low resolution quantization," *Proc. ITC-CSCC,* pp. 175-178, Jul. 1998.

[36] A.K. Ng and B. Zeng, "A new fast motion estimation algorithm based on search window sub-sampling and object boundary pixel block matching," *in Proc. Int. Conf. Image Processing,* pp. 605-608, Oct. 1998.

[37] F.H. Cheng and S.N. Sun, "New fast and efficient two-step search algorithm for block motion estimation," *IEEE Trans. Circuits Syst. Video Technol.,* vol. 9, no. 10, pp. 977-983, Oct. 1999.

[38] K. Lengwehasatit and A. Ortega, "Probabilistic partial-distance fast matching algorithms for motion estimation," *IEEE Trans. Circuits Syst. Video Technol.,* vol. 11, no. 2, pp. 139-152, Feb. 2001.

[39] A. Tourapis, O.C. Au, and M.L. Liou, "Highly efficient predictive zonal

algorithms for fast block-matching motion estimation," *IEEE Trans. Circuits Syst. Video Technol.,* vol. 12, no. 10, pp. 934-947, Oct. 2002.

[40] Y. Naito, T. Miyazaki and I. Kuroda, "A fast full-search motion estimation method for programmable processors with a multiply-accumulator," *Proc. ICASSP*, pp. 3221-3224, 1996.

[41] K.T. Choi, S.C. Chan, and T.S. Ng, "A new fast motion estimation algorithm using hexagonal subsampling pattern and multiple candidates search," *in Proc. Int. Conf. Image Process.,* pp. 497-500, Sep. 1996.

[42] Y.C. Lin and S.C. Tai, "Fast full-search block-matching algorithm for motion-compensated video compression," *IEEE Trans. Commun.*, vol. 45, pp. 527-531, May 1997.

[43] Y.L. Chan and W.C. Siu, "New adaptive pixel decimation for block motion vector estimation," *IEEE Trans. Circuits Syst. Video Technol.,* vol. 6, no. 1, pp. 113-118, Feb. 1996.

[44] Y. Wang, Y. Wang and H. Kuroda, "A globally adaptive pixel-decimation algorithm for block-motion estimation," *IEEE Trans. Circuits Syst. Video Technol.,* vol. 6, no. 10, pp. 1006-1011, Sep. 2000.

[45] N. Roma and L. Sousa, "Efficient and configurable full-search block-matching processors," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, pp. 1160-1167, Dec. 2002.

[46] R. Li, B. Aeng and M.L. Liou, "A new three-step search algorithm for block motion estimation," *IEEE Trans. Circuits Syst. Video Technol.,* vol. 4, pp. 438-442, Aug. 1994.

[47] J.N. Kim and T.S. Choi, "A fast three-step search algorithm with minimum checking points using unimodal error surface assumption," *IEEE Trans. Consumer Electronics,* vol. 44, no. 3, pp. 638-648, Aug. 1998.

[48] J.N. Kim and T.S. Choi, "Reduction of checking points using unimodal error surface assumption for fast motion estimation," *Proc. SPIE*, vol. 3460, San Diego, pp. 124-134, Jul. 1998.

[49] J.N. Kim and T.S. Choi, "An efficient three-step search algorithm using unequal precision of searching," *Proc. ITC-CSCC,* vol. 1, pp. 903-906, Jan. 1998.

[50] X. Jing and L.P. Chau, "An efficient three-step search algorithm for block motion estimation," *IEEE Trans. Multimedia,* vol. 6, no. 2, pp. 435-438, Jun. 2004.

[51] L.M. Po and W.C. Ma, "A novel four-step search algorithm for fast block motion estimation," *IEEE Trans. Circuits Syst. Video Technol.,* vol. 4, pp. 313-317, Jun. 1996.

[52] J.S. Kim and R.H. Park, "A fast feature-based block matching algorithm using integral projections," *IEEE Trans. Commun.,* vol. 10, pp. 968-971, Jun. 1992.

[53] K.M. Uz, M. Vetterli, and D. LeGalll, "Interpolative multiresolution coding of advanced television with compatible subchannels," *IEEE Trans. Circuits Syst. Video Technol.,* vol. 1, pp. 86-99, Mar. 1991.

[54] J. Chalidabhongse and C.C. Jay Kuo, "Fast motion vector estimation

using multiresolution-spatio-temporalcorrelations," *IEEE Trans. Circuits Syst. Video Technol.,* vol. 7, pp. 477-488, Jun. 1997.

[55] K.W. Lim, B.C. Song and J.B. Ra, "Fast hierarchical block matching algorithm utilizing spatial motion vector correlation," *Proc. SPIE,* vol. 3024, pp. 284-292, 1997.

[56] J. Wei and Z.N. Li, "An enhancement to MRMC scheme in video compression," *IEEE Trans. Circuits Syst. Video Technol.,* vol. 7, pp. 564-568, Jun. 1997.

[57] Y.Q. Shi and X. Xia, "A thresholding multiresolution block matching algorithm," *IEEE Trans. Circuits Syst. Video Technol.,* vol. 7, pp. 437-440, Feb. 1997.

[58] A. Wang and M. Ghanbari, "Scalable coding of very high resolution video using the virtual zerotree," *IEEE Trans. Circuits Syst. Video Technol.,* vol. 7, pp. 719-726, Feb. 1997.

[59] X. Lee and Y.Q. Zhang, "A fast hierarchical motioncompensation scheme for video coding using block feature matching," *IEEE Trans. Circuits Syst. Video Technol.,* vol. 6, pp. 627-635, Dec. 1996.

[60] B.C. Song, K.W. Lim, and J.B. Ra, "A fast motion estimation algorithm using spatial correlation of motion field and hierarchical search," *Proc. SPIE,* vol. 2952, pp. 308-315, 1996.

[61] C.K. Cheung and L.M. Po, "A hierarchical block matching algorithm using partial distortion measure," *Proc. ISCAS,* pp. 1237-1240, 1997.

[62] J.Y. Tham, S. Ranganath, M. Ranganth, and A. A. Kassim, "A novel

unrestricted center-biased diamond search algorithm for block motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 8, no. 4, pp. 369-377, Aug. 1998.

[63] S. Zhu and K.K. Ma, "A new diamond search algorithm for fast blockmatching motion estimation," *IEEE Trans. Image Processing,* vol. 9, no. 2, pp. 287-290, Feb. 2000.

[64] C.H. Cheung and L.M. Po, "A novel cross-diamond search algorithm for fast block motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, no. 12, pp. 1168-1177, Dec. 2002.

[65] Y. Nie and K.K. Ma, "Adaptive road pattern search for fast block matching motion estimation," *IEEE Trans. Image Processing,* vol. 11, no. 12, pp. 1442-1449, Dec. 2002.

[66] C. W. Lam, L. M. Po and C. H. Cheung, "A novel kite-cross-diamond search algorithm for fast block matching motion estimation," *in Proc. IEEE Int. Symp. Circuits and Syst.,* vol. 3, pp. 729-732, May 2004.

[67] Z. Chen, Y. He and J. Xu, "Hybrid unsymmetrical-cross multihexagon-grid search strategy for integer pel motion estimation in H.264," *in Proc. Picture Coding Symp.*, Saint Malo, pp. 17-22, Apr. 2003.

[68] C.H. Cheung and L.M. Po, "Novel cross-diamond-hexagonal search algorithms for fast block motion estimation," *IEEE Trans. Multimedia,* vol. 7, no. 1, pp. 16-22, Mar. 2005.

[69] C. Zhu, X. Lin, L.P. Chau, and L.M. Po, "Enhanced hexagonal search

for fast block motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 14, no. 10, pp. 1210-1214, Oct. 2004.

[70] X. Yi and N. Ling, "Zero-motion-vector-biased cross-diamond search algorithm for rapid matching motion estimation," *in Proc. Image and Video Communications and Processing 2005,* pp. 995-1006, 2005.

[71] C. Zhu, X. Lin, and L.P. Chau, "Hexagon-based search pattern for fast block motion estimation," *IEEE Trans. Circuits Syst. Video Technol.,* vol. 12, no. 5, pp. 349-355, May 2002.

[72] C. Zhu, X. Lin, and L.P. Chau, "Enhanced Hexagon search for fast block motion estimation," *IEEE Trans. Circuits Syst. Video Technol.,* vol. 14, no. 10, pp. 1210-1214, Oct. 2004.

[73] W. Li and E. Salari, "Successive elimination algorithm for motion estimation," *IEEE Trans. Image Processing*, vol. 4, pp. 105-107, Jan. 1995.

[74] X.Q. Gao, C.J. Duanmu, and C.R. Zou, "A multilevel successive elimination algorithm for block matching motion estimation," *IEEE Trans. Image Processing*, vol. 9, pp. 501-504, Mar. 2000.

[75] J.N. Kim and T.S. Choi, "A fast motion estimation for software based real-time video coding," *Proc. ISCE,* pp. TPB1-16 - TPB1-21, Oct. 1998.

[76] J.N. Kim and T.S. Choi, "Adaptive matching scan algorithm based on gradient magnitude for fast full search in motion estimation," *IEEE Trans. Consumer Electronics*, vol. 45, pp. 762-772, Aug. 1999.

[77] J.N. Kim and T.S. Choi, “A fast full-search motion-estimation algorithm using representative pixels and adaptive matching scan,” *IEEE Trans. Circuits Syst. for Video Technol.*, vol. 10, pp. 1040-1048, Oct. 2000.

[78] T.M. Oh, Y.R. Kim, W.G. Hong, and S.J. Ko, “A fast full search motion estimation algorithm using the sum of partial norms,” *Proc. ICCE*, pp. 236-237, 2000.

[79] J.N. Kim, S.C. Byun, Y.H. Kim, and B.H. Ahn, “Fast Full Search Motion Estimation Algorithm Using Early Detection of Impossible Candidate Vectors”, *IEEE Trans. Signal Processing*, vol. 50, pp. 2355-2365, Sep. 2002.

[80] T.K. Ryu, K.W. Kang, G.S. Jung, and J.N. Kim, “A Fast Elimination Algorithm of Candidate Vectors Using Adaptive Matching Scan” *WIAMIS2006,* pp. 153-156, Apr. 2006.

[81] T.K. Ryu, G.S. Jung, and J.N. Kim, “A Fast Full Search Algorithm for Motion Estimation Using Priority of Matching Scan” *LNCS* vol. 4141, pp. 571-579, Aug. 2006.

[82] J.N. Kim, T.K. Ryu, and Y.J. Jeong, “A Fast Partial Distortion Elimination Algorithm Using Selective Matching Scan”, *LNCS* vol. 4263, pp. 125-133, Nov. 2006.

[83] 유태경, 황진호, 문광석, 김종남, “최초 부분계수 비교를 통한 후보 벡터의 고속 제거 알고리즘” 한국멀티미디어학회 춘계학술대회 논문집, vol. 9, no. 1, pp. 655-659, 2006.

[84] 유태경, 정용재, 정태일, 문광석, 김종남, “최초 부분계수 비교를 통한 고속 PDE 알고리즘,” 한국신호처리시스템학회 하계학술대회논문집, vol. 8, no. 1, pp. 70-73, 2007.

[85] T.K. Ryu, T.I. Jeong, K.Y. Ryu, K.S. Moon, and J.N. Kim, “A Fast Partial Distortion Elimination Algorithm Using Adaptive Matching Scan and Refining Threshold,” *CCIS,* vol. 2, pp. 801-809, Aug. 2007.

[86] 유태경, 정용재, 정태일, 문광석, 김종남, “부분 임계치 적응예측을 통한 고속 PDE 알고리즘,” 한국멀티미디어학회 추계학술발표대회논문집, vol. 10, no. 2, pp. 419-422, Nov. 2007.

[87] C.K. Cheung and L.M. Po, “Normalized partial distortion search algorithm for block motion estimation,” *IEEE Trans. Circuits Syst. for Video Technol.,* vol. 10, pp. 417-422, Apr. 2000.

[88] C.K. Cheung and L.M. Po, “Adjustable partial distortion search algorithm for fast block motion estimation,” *IEEE Trans. Circuits Syst. for Video Technol.*, vol. 13, no. 1, pp. 100-110, Jan. 2003.

[89] B. Montrucchio and D. Quaglia, “New sorting-based lossless motion estimation algorithms and a partial distortion elimination performance analysis,” *IEEE Trans. Circuits Syst. Video Technol.,* vol. 15, no. 2, pp. 210-220, Feb. 2005.

[90] JM Reference Software, [Online]. Available: http://bs.hhi.de/suehring/tml/download/

[91] G. Bjontegaard and K. Lillevold, “Context-adaptive VLC coding of coefficients,” in JVT-C028, Joint Video Team (JVT) of ISO/IEC MPEG

& ITU-T VCEG, 3rd Meeting, Fairfax, VA, May 2002.

[92] X. Yi, J. Zhang, N. Ling, and W. Shang, “Improved and simplified fast motion estimation for JM,” in JVT-P021, Joint Video Team of ISO/IEC MPEG & ITU-T VCEG, 16th Meeting, Poznan, Poland, Jul. 2005.

[93] J. Zhang, X. Yi, N. Ling, and W. Shang, “Context adaptive Lagrange multiplier (CALM) for motion estimation in JM-improvement,” in JVT-T046, Joint Video Team of ISO/IEC MPEG & ITU-T VCEG, 20th Meeting, Klagenfurt, Austria, Jul. 2006.

감사의 글

이 논문이 완성되기까지 정성어린 지도와 편달을 아끼지 않으시고 용기를 주신 문광석 교수님께 진심으로 감사드립니다. 대학원생활에 있어서 적절할 지적과 충고를 해 주신 김종남 교수님께 깊은 감사를 드리며, 여러 가지로 부족한 본 논문의 심사를 맡아 세심한 조언을 해주신 권태하 교수님, 권기룡 교수님, 류권열 교수님 그리고 학위과정동안 많은 가르침을 주셨던 학과의 여러 교수님들께도 감사드립니다.

박사 과정 동안 같은 연구실에서 실험실생활에 물심양면으로 도움을 주신 류권열 교수님, 허영대 박사님, 정태일 박사님, 서호찬 선생님께 감사드립니다. 지난 기간 동안 같은 연구실에서 동고동락하며 세심한 배려와 따가운 충고를 해준 경원 형, 친구 용재, 그리고 원희, 진호 그리고 논문을 준비하면서 도와준 나머지 연구실 후배들에게도 감사 드립니다. 학교 생활에 있어 많은 도움을 준 태희선배, 수진선배, 봉기선배, 윤귀씨에게도 고마움을 전합니다.

늘 부족한 아들 뒷바라지를 해주신 어머니와 늘 자식 걱정이 앞서는 아버지께 오늘의 영광을 돌리고 싶습니다. 그리고 동생 길미 부부, 지민에게도 감사의 말을 전합니다. 늘 사위를 믿어주시는 장인 장모님께도 고마움을 전합니다. 마지막으로 이 세상에서 가장 사랑하는 아내 세은이와 함께 이 기쁨을 나누고 싶습니다.

2008년 2월

류 태 경 올림