Journal of Korea Multimedia Society Vol. 8, No. 3, March 2005(pp. 345-354)

실시간 영상에서 반복적인 움직임에 적응한 블록정합 알고리즘 설계

강진석[†], 김장형[††]

요 약

실시간 동영상에서의 프레임간의 상관관계를 이용하여 움직임 추정과 움직임 보상기법을 수행한다. 이 방법은 동영상에 존재하는 중복된 데이터를 제거하기 때문에 실시간 영상에서 중요한 역할을 하지만 실시간 응용 및 고해상도 응용에 적용하기에는 많은 계산량을 필요로 한다. 따라서 움직임 벡터를 결정하기 전에 블록 내부의 움직임을 예측한다면, 이를 바탕으로 탐색 영역에서 초기 탐색점 위치와 탐색 패턴을 결정할 수 있을 것이다. 본 논문에서는 차영상을 통해 움직임의 검출 되었을 때, 이 움직임의 물체의 반복적인 영상인지, 아니면 침입자가 발생한 영상인지를 구분하였으며 움직임 검출 영역 간 픽셀 값의 분포도를 나타낸다. 제안된 알고리즘은 움직임 보상 예측된 화질 및 계산량의 감소에 있어서 높은 성능 향상을 보였다.

The Design of Repeated Motion on Adaptive Block Matching Algorithm in Real-Time Image

Jin-Suk Kang[†], Jang-Hyung Kim[††]

ABSTRACT

Since motion estimation and motion compensation methods remove the redundant data to employ the temporal redundancy in images, it plays an important role in digital video compression. Because of its high computational complexity, however, it is difficult to apply to high-resolution applications in real time environments. If we have a priori knowledge about the motion of an image block before the motion estimation, the location of a better starting point for the search of an exact motion vector can be determined to expedite the searching process. In this paper presents the motion detection algorithm that can run robustly about recusive motion. The motion detection compares and analyzes two frames each other, motion of whether happened judge. Through experiments, we show significant improvements in the reduction of the computational time in terms of the number of search steps without much quality degradation in the predicted image.

Key words: Block Matching(블록정합), Difference Image(차영상), MAE(평균절대오차)

1. 서 론

영상 감시 시스템에서 움직임을 검출하는 방법에는 크게 프레임 간 차이법과 배경 차이법으로 나눌 수 있다. 프레임 간 차이법은 현재 입력된 입력 영상과 바로 직전에 입력된 영상이 픽셀 값들을 서로 비교 분석하여 움직임을 검출하는 방법이고 배경 차이법은 현재 입력된 영상과 이전에 입력된 모든 영상들

※ 교신저자(Corresponding Author) : 강진석, 주소 : 전라북도 군산시 미룡동 산68(573-701), 전화 : 063)469-4113, FAX : 063)469-4699, E-mail: jskang01@kunsan.ac.kr
접수일 : 2004년 9월 17일, 완료일 : 2004년 11월 4일
[†] 정회원, 군산대학교 전자정보공학부 BK계약교수
[††] 정회원, 제주대학교 통신컴퓨터공학부 교수
(E-mail : janghhk@cheju.ac.kr)
※ 본 논문은 군산대학교 두뇌한국 21의 일부지원에 의해 수행되었음.

이 평균값을 구하여 배경 영상을 만들고 현재 영상과 비교 분석하여 움직임을 검출해 내는 방법이다. 이러한 여러 가지 움직임 검출 기법들 중 현재 가장 많이 사용되는 움직임 검출 기법에는 차영상 움직임 검출 기법이 있다. 차영상 움직임 검출 기법은 기존의 다양한 움직임 검출 기법 중 가장 처리속도가 빠르고, 알고리즘 구현이 쉽기 때문에 실제 많이 사용되는 움직임 검출 기법이다. 하지만 이 기법은 입력 장치에 유입되는 잡음이나 반복적인 움직임을 갖는 물체의 동작에 상당히 민감하게 동작 함으로써 신뢰성 있는 움직임 검출이 어렵다는 문제점을 가지고 있다.

따라서 본 논문에서는 이러한 차영상 움직임 검출 기법의 문제점을 개선하기 위해 반복적인 물체의 움직임을 검출하는 알고리즘을 제시 하고자 한다. 제시한 알고리즘은 먼저 배경이 되는 이미지와 현재 이미지의 차를 이용하여 현재 움직임의 발생 하였는지 판단하고, 움직임의 발생 하였을 경우 두 이미지간 차영상(Difference Image)을 이진화 처리하여, 움직임의 발생한 영역을 블록 영역을 정합(Matching) 시킨다. 그리고 나서 두 블록 영상 간에 평균절대오차(Mean Absolute Error : MAE) 값을 구하여 평균절대오차 값이 작으면 반복된 물체의 움직임으로 판단하고, 반대로 값이 크면 새로운 움직임으로 판단하도록 하였다. 이때 반복된 물체의 움직임인 경우에는 배경이 되는 영상을 현재 입력된 영상으로 갱신하고, 새로운 움직임인 경우에는 갱신되지 않도록 하였다.

2. 움직임 검출 기법

움직임 검출은 입력되는 3차원의 실세계의 영상을 2차원 데이터로 나타내고 이 데이터에서 움직임이 있는 부분만을 영역화 하여 표현 가능하다. 즉 부분 영역화란 입력되는 정보로부터 동일한 특징을 갖는 영역으로 구분하는 과정을 의미한다. 주로 영상에서 사용되는 움직임 검출 기법은 움직임이 있는 영상 정보를 일정한 시간 간격으로 입력받아 입력된 두 영상을 서로 비교하여 영상 정보의 차를 조사, 분석하여 움직임으로 판단하는 방법을 사용한다. 즉 입력된 영상의 각 프레임들을 서로 비교 분석하여 움직임이 발생한 영역을 찾아내고, 그 영역의 특징들을 이용하여 움직임을 인식하거나 정의한다.

2.1 블록 정합 알고리즘

블록 정합 알고리즘(Block Matching Algorithm)은 원래 디지털 영상 신호전송에서 프레임 간 중복성을 줄이기 위한 움직임 추정 방법으로 현재 영상 프레임과 이전 영상 프레임의 제한된 후보 영역(Candidate Area)들 중에서 최적의 정합점(Matching Point)을 찾아내는 방법이다. 일반적으로 각 블록은 중첩되지 않은 상태에서 대상을 인식하여 대상 블록이 새로운 위치를 나타내는 변위 벡터(Displacement Vector)를 구하게 된다. 이때 블록이 크기가 커지면 이전 프레임에서 일치하는 블록을 찾는 탐색 시간은 줄어들고 영상의 화질은 떨어지게 된다.

블록 정합이 주 원리는 현재 프레임의 블록과 이전 프레임 중, 후보 블록을 선정하고 정합 함수(Matching Function : MF)를 계산하는 것이다. 이 과정을 후보 블록 전체에 대해 반복한 후 가장 정합이 잘된 블록을 결정하고 여기서 결정된 블록의 위치와 이동전 위치까지의 거리와 방향의 추정된 변위 벡터이다. 만약 화상의 블록크기가 N×M이고, 최대 수평, 수직 방향이 변위량을 각 d_{mzx-x}, d_{max-y} 라고 하면 모든 탐색 영역(Search Region)의 크기는 그림 1처럼 (N+$2dmzx-x$)×(M+$2d_{max-y}$)가 된다. 이때 가능한 탐색점이 수는 총 $(2d_{mzx-x}+1)\times(2d_{max-y}+1)$이 된다.

불록 정합을 위해 많이 사용되는 정합함수(Matching Function : MF)로는 평균절대오차(Mean Absolute Error : MAE), 상호상관함수(Cross Correlation Function : CCF), 평균자승오차(Mean Squared Error : MSE), 최소화된 최대오차(Minimized Maximun Error Function : MME) 등의 있다. 이들 중 계산량

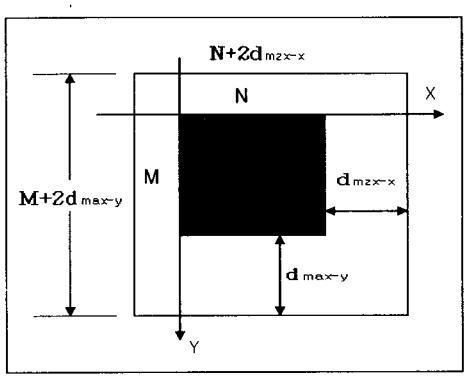


그림 1. 블록정합 탐색 영역

이 적어 알고리즘 구현이 용이한 평균절대오차(MAE)가 가장 많이 사용되고 있다. 다음 식(1)과 식(2)는 각각 평균절대오차(MAE)와 평균자승오차(MSE)를 구하는 식을 나타내고 있다. 여기서 NM은 후보블록의 크기를 나타내고, f_t와 f_{t-1}은 각각 현재 프레임의 후보 블록과 이전 프레임의 블록을 나타낸다.

$$MAE(d_1,d_2)=\frac{1}{NM}\sum_{n=0}^{N-1}\sum_{m=0}^{M-1}|f_t(n,m)-f_{t-1}(n-d_1,m-d_2)| \quad (1)$$

$$MSD(d_1,d_2)=\frac{1}{NM}\sum_{n=0}^{N-1}\sum_{m=0}^{M-1}[f_t(n,m)-f_{t-1}(n-d_1,m-d_2)]^2 \quad (2)$$

2.1.1 전역 탐색 알고리즘

블록 정합(Block Matching)에서 이동 가능한 최대 거리가 각각 d_{max-x} 및 d_{max-y}이므로 현재 프레임의 블록과 같은 좌표를 가지는 이전 프레임의 블록을 중심으로 탐색 영역은 수평, 수직 방향으로 N과 M 만큼씩 이동시키면서 탐색 범위내의 가능한 모든 블록을 조사하여 평균 절대 오차(MAE)를 계산하는 방법이다. 이러한 전역 탐색 알고리즘은 탐색 범위 내에서 가장 적합한 소프트웨어 구현에 많은 어려움을 가지고 있어 이를 해결하기 위해 여러 가지 고속 블록 정합 알고리즘(Fast Block Matching Algorithm : FBMA)들을 사용한다.

2.1.2 고속 탐색 알고리즘

전역탐색 알고리즘 자체가 계산 량이 많아 실제 구현에 사용할 수 없는 문제점을 해결하기 위해 나온 블록 정합 알고리즘(Block Matching Algorithm)으로 대표적인 고속 탐색 알고리즘(FSA)으로는 3단계 탐색 알고리즘(Three Step Search Algorithm : TSS), 2D-log 탐색 알고리즘(2 Dimension LOGarithmic Search Algorithm : 2D-LOG), 4단계 탐색 알고리즘(Four Step Search Algorithm : 4SS), 2단계 탐색 알고리즘(2 Step Search Algorithm : 2SS), 다이아몬드 탐색 알고리즘(Diamond Search Algorithm : DS)등이 있다. Koga에 의해 제안된 3단계 탐색 알고리즘은 가장 고속 정합이 가능한 알고리즘으로서, 탐색 영역의 반의 크기로 탐색 영역의 중심에서 시작하여 각 단계마다 앞 단계 탐색 범위의 반으로 줄여가면서 반복 계산하여 이동 벡터를 구하는 방법이다. 하지만 이 알고리즘은 정합 속도는 빠르지만, 정밀한 정합을 얻을 수 없다는 단점이 있다. A. K. Jain이 제안한 2D-log 탐색법은 이진 탐색(Binary Search)를 확장한 형태로서 블록의 좌상 화소를 기준으로 하여 움직임 예측의 변위화소 길이만큼 떨어진 4개의 점에 대한 평균 절대 오차(MAE) 값을 구하고 이를 5개의 점에서 평균 절대 오차(MAE) 값의 가장 작은 값의 위치를 중심으로 하여 1단계 과정을 반복한다. 이때 탐색 영역의 경계선에 도달하거나 최소점이 중심에서 나타나면 변위화소 길이를 반으로 줄어서 1단계 과정을 반복한다. 이러한 과정을 탐색 간격이 1이 될 때까지 반복한 후 이의 최종 길이를 이동 벡터로 간주하는 방법이다.

그림 2는 각각 3단계 탐색 알고리즘(3SS), 2D-log 탐색 알고리즘(2D-LOG)의 움직임 추정 과정을 보여주고 있다.

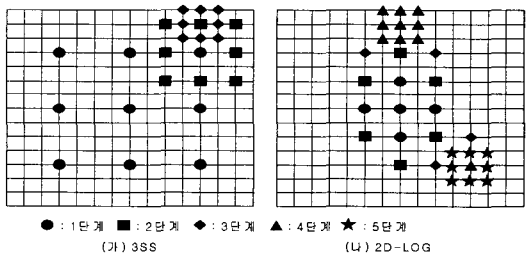


그림 2. 검색 알고리즘

3. 블록 매칭 알고리즘 설계

본 논문에서 입력 영상에서 들어온 첫 번째 장면에서 키 프레임을 추출하고 추출된 키 프레임 컬러 영상을 256 gray 영상으로 바꾸어 다른 키 프레임과 비교할 배경 키 프레임으로 지정하고 두 번째부터 들어온 프레임을 키 프레임을 설정하여 이를 그레이 영상으로 변화시켜 지정된 배경 키 프레임(Background Key Frame)과의 차영상(Difference Image)을 구한다. 이렇게 차영상을 구하고 난 후 블록정합을 위해 먼저 차영상을 이진화 처리하고 이진화된 차영상을 가지고 블록 영역(Block Area)좌표를 추출한다. 그 다음 블록 영역의 (x, y) 필셀 좌표를 지정된 배경과 현재 키 프레임에 각각 적용시켜 블록 영역을 설정한다. 다음 설정된 두 블록 영역은 각 픽셀 좌표간 블록정합시켜 평균절대오차(Mean Absolute Error : MAE)를 계산한다. 이때 계산된 평균절대오차가 임계값(Threshold) 보다 작으면 반복적인 물체의 움직

임으로 간주하여 침입자가 발생하지 않은 것으로 판단하고 만약에 크면 침입자가 발생한 것으로 간주되어 현재 입력되어 있는 영상은 저장되도록 하였다. 또한, 평균절대오차 값이 설정한 임계값 보다 크면, 배경 키 프레임을 현재 입력된 영상의 키 프레임으로 갱신하여 메모리에 저장해두고, 만약 작으면 새로운 움직임의 발생한 것으로 판단하여 배경이 되는 키 프레임의 갱신되지 않고 이전 키 프레임이 계속 유지되도록 하였다. 다음 그림 3은 본 논문에서 제안한 시스템의 전체 구성도를 보여주고 있다.

그림 3. 시스템의 블록도

3.1 차영상 획득과 이진화 처리

입력 장치를 통해 입력된 컬러 영상을 256 그레이 흑백 영상으로 변환하는 전처리 과정을 거친 후에는 배경 키 프레임(Key Frame)과 현재 입력된 영상에서 추출한 키 프레임을 이용해 두 프레임 간에 차영상(Difference Image)을 구해야 한다. 이때 배경 키 프레임은 영상 입력 장치를 통해 가장 먼저 입력된 영상에서의 키 프레임을 배경 키 프레임으로 설정한다. 차영상을 획득하는 식(3)와 같다.

$$DI(x,y) = |g(x,y) - g_b(x,y)|$$
$$DI(x,y) = \begin{cases} g(x,y) & (DI(x,y) > t) \\ 0 & (DI(x,y) \le t) \end{cases} \quad (3)$$

식(3)에서 $g(x,y)$는 현재 입력된 컬러 영상의 256 그레이 흑백 영상으로 변환된 후, 키 프레임을 표현하며 $g_b(x,y)$는 배경의 되는 키 프레임으로 $DI(x,y)$는 현재 입력 영상 키 프레임과 배경 키 프레임간 픽셀들의 차를 통해 획득된 차영상을 각각 나타낸다. 이때 차영상의 픽셀값이 지정된 임계값 t 보다 크면, 현재 입력된 영상의 키 프레임의 픽셀 값을 할당하고 만약 작거나 같은며 0 값을 할당한다. 그림 4는 두 키 프레임간에 차영상을 구한 결과를 보여주고 있다.

이때 검출된 차영상은 블록정합 처리를 하기 위한 전처리 과정으로 정합시킬 블록 영역(Block Area)을 설정하기 위해 먼저 영상의 픽셀값을 0과 255로 지정하는 이진화 처리를 수행한다. 다음 식 (4)과 (5)은 각각 차영상을 이진화 처리하기 위해 임계값(Threshold)을 설정하고 처리하는 수식을 보여주고 있다.

(a) (b) (c)

그림 4. 배경 영상(a), 현재 영상(b), 차 영상(c)

$$T = \sum_{x=b}^{x=N} \sum_{y=b}^{y=M} DI(x,y)/(NM) \quad (4)$$

$$DI(x,y) = \begin{cases} 255 & (DI(x,y) > T \\ 0 & (DI(x,y) \le T \end{cases} \quad (5)$$

위 식 (4)과 (5)에서 $DI(x,y)$는 차영상을 나타내고 있고 T는 임계값, NM은 차영상의 크기를 각각 나타내고 있다. 이때 임계값 T는 차영상 $DI(x,y)$의 모든 픽셀값들의 평균을 T로 설정하여 이진화 처리를 하게 된다.

3.2 블록 정합(Block Matching)

본 논문에서는 먼저 차영상을 이진화 처리하여 블록정합을 적용시킬 블록 영역의 범위를 설정한다. 즉 블록 매칭 알고리즘를 적용하여 평균절대오차(MAE)를 검출할 블록영역의 범위는 움직임의 발생한 영역만을 범위로 설정한다.

블록 영역은 이진화 처리된 차 영상들을 행과 열로 한 줄씩 차례대로 검색하여 255의 픽셀값을 갖는 픽셀들의 개수를 먼저 구한다. 일반적으로 이진화 처리된 차영상에서 많은 움직임의 발생한 영역은 255 픽셀 값의 빈도수가 비교적 많다. 이를 이용하여 255 값을 갖는 픽셀들의 빈도수를 행과 열로 한 줄씩 검색하여 임계값 보다 큰 x, y 좌표값 들을 먼저 구한다. 이렇게 구한 x, y 좌표 값들 중 최소, 최대 값을 가지는 x, y 값들을 각각 $x_{\min}$, $x_{\max}$, $y_{\min}$, $y_{\max}$ 값으로 지정하고, 이 4개의 값을 이용하여 $(x_{\min}, y_{\min})$, $(x_{\min}, y_{\max})$, $(x_{\max}, y_{\min})$, $(x_{\max}, y_{\max})$를 좌표로 하는 사각형 영역을 블록 영역으로 설정한다. 이진화 처리된 차영상에서 $x_{\min}, x_{\max}, y_{\min}, y_{\max}$ 좌표 값을 구하는 알고리즘은 다음과 같다.

Step. 1 : x축으로 차영상의 픽셀값들의 수를 한줄씩 차례대로 검색한다.

Step. 2 : 한줄 검색의 끝나면 255값을 가지는 픽셀의 개수를 구한다.

Step. 3 : 255값을 가지는 픽셀들의 개수가 임계값 이상이면 이때 y좌표의 값을 저장한다.

Step. 4 : x축 검색이 끝나면 저장된 y좌표값들 중에서 최소값을 $y_{\min}$으로, 최대값을 $y_{\max}$ 값으로 각각 지정한다.

Step. 5 : $y_{\min}$값과 $y_{\max}$값을 구한 후 Step. 1부터 Step. 4까지 y축으로 똑같은 방법을 적용하여 $x_{\min}, x_{\max}$ 값을 각각 구한다.

다음 그림 5는 $x_{\min}, x_{\max}, y_{\min}, y_{\min}$ 좌표를 가지고 블록 영역을 설정하는 모습을 보여 주고 있다.

이처럼 블록 영역을 설정한 후에는 실제 설정된 블록 영역의 좌표를 실제 블록 정합(Block Matching)시킬 프레임에 대입해야 한다. 즉 $(x_{\min}, y_{\min})$, $(x_{\min}, y_{\max})$, $(x_{\max}, y_{\min})$, $(x_{\max}, y_{\max})$ 좌표를 메모리에 저장해둔 배경키 프레임과 현재 영상 키 프레임에 각각 설정된 4개의 좌표값을 대입시켜 실제 블록 정합(Block Matching)시킬 두 개의 블록을 얻는다.

그림 6의 (a), (b)는 이진화 처리된 차영상에서 $(x_{\min}, y_{\min})$, $(x_{\min}, y_{\max})$, $(x_{\max}, y_{\min})$, $(x_{\max}, y_{\max})$ 좌표를 구하여 이를 기준으로 하는 최대 사각형 영역을 설정하여 블록 영역을 검출한 결과 영상을 보여주고 있다. 그림 6의 (c)는 실제 블록 영역 좌표를 배경과 현재 영상키 프레임에 각각 대입하여 배경 블록과 현재 블록을 얻어온 모습을 보여주고 있다.

이때 지정된 두 개의 블록을 식(6)를 이용해 블록 정합시켜 평균절대오차를 계산하고, 계산된 값을 가지고 물체의 반복적인 움직임인지 아니면 침입자가 발생한 움직임 인지를 구분하게 된다. 다음 식(6)에서 NM은 블록 영역의 크기를 나타내고 $f_t(n,m)$은 배경 블록을 $f_{t-1}(n-d_1, m-d_2)$은 현재 블록을 각각 나타내고 있다.

$$MAE(d_1,d_2)=\frac{1}{NM}\sum_{n=0}^{N-1}\sum_{m=0}^{M-1}|f_t(n,m)-f_{t-1}(n-d_1,m-d_2)| \quad (6)$$

실제 물체의 반복적인 움직임인 경우에는 배경 키 프레임과 현재 영상 키 프레임간, 픽셀 값의 분포도가 비슷함으로 인해 계산되는 평균 절대오차 값은 작고, 반대로 침입자가 발생한 영상에서는 배경 키 프레임과 현재 영상 키 프레임 간, 픽셀 값의 분포도가 크게 차이가 나기 때문에 계산되는 평균절대오차 값은 커지게 된다. 따라서 계산된 평균절대오차 값이

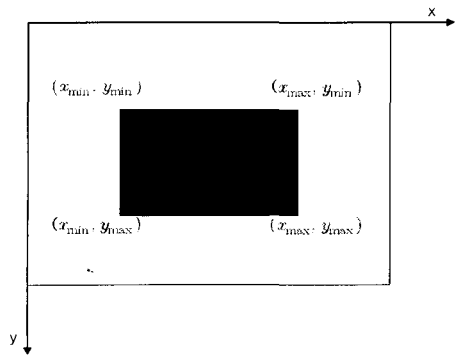
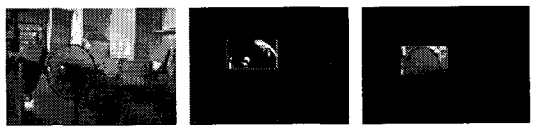


그림 5. 블록 영역



(a) (b) (c)

그림 6. 블록 영역 검출

특정 임계 값보다 작으면 이때 입력된 영상은 침입자가 아닌 물체의 반복적인 움직임으로 검출하고, 반대로 크면 이는 침입자가 발생한 영상으로 판단하고 입력되는 영상들은 감시 시스템에 저장되기 시작한다.

3.3 배경 키 프레임 갱신

앞서 배경 블록과 현재 블록 간 블록정합을 시키고, 그때 계산된 평균절대오차(MAE)값을 분석하여 물체의 반복적인 움직임의 발생한 영상과 침입자가 발생한 영상을 구분할 수 있도록 하였다. 이때 물체의 반복적인 움직임의 발생한 영상인 경우에는 배경 키 프레임을 현재 입력 영상 키 프레임으로 갱신한다. 다음 식(7)은 물체의 반복적인 움직임인지 판단하여 배경 키 프레임을 갱신하는 수식을 보여주고 있다. 여기서 $g(x,y)$는 현재 입력 영상 키 프레임이고, $g_b(x,y)$는 배경 키 프레임을 각각 나타내고 있으며, t 값은 평균절대오차(MAE)값과 비교할 임계값을 나타내고 있다.

$$g_b(x,y)=\begin{cases} g(x,y) & (MAE < t) \\ g_b(x,y) & (MAE \le t) \end{cases} \quad (7)$$

즉 계산된 평균절대오차 값이 시스템에 설정한 임계값(Threshold : t)보다 작은 경우에만 최근에 입력 영상의 키 프레임으로 배경 키 프레임이 갱신된다.

4. 실험 설계 및 구현 분석

본 논문에서 제안한 반복적인 움직임 검출 기법은 침입자가 발생한 영상과 어떤 물체가 주변 환경에 인해 자연적으로 움직임의 발생한 영상(반복적인 움직임)을 구분하여 침입자가 발생한 영상만 감시 시스템에 저장 될 수 있도록 하는데 목적이 있다.

이 장에서는 반복적인 움직임 검출 기법의 효용성을 보이기 위해 제안한 알고리즘을 구현하여 실제 가장 많이 구현되어 사용되고 있는 차영상 움직임 검출 기법과 비교하여 실험을 하고 그 결과를 비교 분석 하였다. 또한, 본 논문에서 구현한 반복적인 움직임 검출 시스템은 크게 영상획득, 전처리, 움직임 블록정합 및 평균절대오차 계산, 움직임 검출 및 배경 키 프레임 갱신 부분으로 구성된다.

4.1 영상획득 및 전처리

본 논문에서는 움직임이 있는 영상에서의 움직임 검출을 실험하기 위해 카메라를 통해 입력된 320×240 크기의 RGB 컬러 영상을 식(8)을 이용해 256 Gray 레벨의 흑백 영상으로 변환하여 다음과 같은 3가지 종류의 그룹으로 나눈 입력 영상들을 가지고 10번에 걸쳐 실험 하였다.

$$g(x,y) = 0.333R(x,y) + 0.333G(x,y) + 0.333B(x,y) \quad (8)$$

① 입력 영상 1 : 침입자가 발생한 영상
② 입력 영상 2 : 상하, 좌우로 반복 움직임 영상
③ 입력 영상 3 : 불규칙적인 반복 움직임 영상

첫 번째 조건인 입력 영상 1은 침입자가 실제 발생하여 감시 시스템에 저장되어야 할 영상의 배경 프레임과 현재 프레임 상태이며, 두 번째 조건인 입력 영상 2는 침입자가 발생하지 않고 선풍기가 좌우로 회전하여 움직임만 발생한 상태이며, 마지막으로 바람에 의해 종이가 날리면서 움직임의 발생한 영상으로 실제 침입자가 발생하지 않은 경우의 상황에서 실험하였다.

4.2 2진화 처리

그림 7은 각각 앞에서 제시한 실험 입력영상 1, 2, 3에 대한 차영상(Difference Image)을 구현 결과 영상과 이진화 처리한 결과 영상을 각각 보여주고 있다.

(a) 차영상(Difference)

(b) 이진 영상(Binary Image)

그림 7. 차영상(a)과 이진영상(b) 결과 화면

여기서 차영상 움직임 검출 기법인 경우에 그림 7처럼 차영상 만을 구하고 이 차영상을 시스템에서 설정한 임계값과 비교하여 움직임을 검출하게 된다. 하지만 본 논문에서는 블록정합(Block Matching)을 시켜 평균절대오차(MAE)를 계산하기 위한 전처리 과정으로 차영상을 구한 후 다시 한번 영상의 픽셀 값을 0과 255로 설정하는 이진화 처리를 하게 된다.

4.3 블록 영역 검출

이진화 처리된 영상은 실제 블록 정합을 시키기 위해 현재 입력 영상키 프레임과 배경 키 프레임에 대입할 블록 영역을 설정해야 한다. 이때 블록 영역은 이진화 처리된 영상에서 행과 열로 한줄씩 검색하여 255값을 가지는 픽셀들의 계수가 최대, 최소인 x, y값들을 구하고,
$(x_{\min}, y_{\min})$, $(x_{\max}, y_{\min})$, $(x_{\min}, y_{\max})$, $(x_{\max}, y_{\max})$
4개의 좌표값을 블록영역의 좌표값으로 설정하여 이를 메모리에 저장해둔 비경 키 프레임(Background Key Frame)과 현재 영상 키 프레임(Current Key Frame)에 각각 대입시켜 실제 블록정합(Block Matching)시킬 두 개의 블록을 얻는다. 다음은 실험 입력 영상 1, 2, 3에 대해 산출한 블록 영역좌표 표 1과 이 좌표들을 실제 배경 키 프레임(Background Key Frame)과 현재 입력 영상 키 프레임에 적용시켜 블

표 1. 블록 영역 좌표

	$x_{\min}$	$x_{\max}$	$y_{\min}$	$y_{\max}$
입력영상 1	158	210	97	239
입력영상 2	157	294	22	166
입력영상 3	110	185	27	185

록 정합(Block Matching) 시킬 배경 블록과 현재 블록을 얻어온 화면을 각각 보여주고 있다.

4.4 평균절대오차 계산

일반적으로 현재 영상 키 프레임과 배경 키 프레임에서 움직임 물체를 검출하는 방법은 현재 영상 키 프레임에서 움직임 물체의 블록과 가장 유사한 블록을 배경 키 프레임에서 검색하는 방법이다. 본 논문에서는 블록 정합시 두 블록 영역 간(배경 블록과 현재 블록) 매칭의 척도가 되는 함수들 중 평균절대오차(Mean Absolute Error : MAE)을 사용하여 실험하였다. 다음 식(9)는 평균절대오차(MAE)를 구하는 식을 보여주고 있다.

$$MAE(d_1,d_2)=\frac{1}{NM}\sum_{n=0}^{N-1}\sum_{m=0}^{M-1}|f_t(n,m)-f_{t-1}(n-d_1,m-d_2)| \quad (9)$$

위 식에서 f_t는 현재 영상 키 프레임이고 f_{t-1}은 배경 키 프레임이며 NM은 블록영역(Block Area)의 크기를 각각 나타내고 있다. 표 2는 각 입력 영상들의 현재 입력 영상 키 프레임과 배경 영상 키 프레임을 전처리 과정(Pre-processing)만을 처리한 후 평균절대오차(MAE)값을 계산한 결과의 차영상을 구하고 이진화 처리 후 블록 영역을 구하고, 이 블록 영역 좌표를 현재 영상 키 프레임과 배경 키 프레임에 대입하여 추출된 현재 블록과 배경 블록을 가지고 평균절대오차(MAE)값을 계산하여 실험한 결과이다.

표 2를 분석해 보면 블록 영역을 설정하지 않고 256 그레이 영상으로만 변환하는 전처리 과정만을 거친 후 평균절대오차(MAE)값을 계산해 본 결과, 침입자가 발생한 실험 입력 영상 1과 반복적인 움직임을 갖는 실험 입력 영상 2와 입력 영상 3는 그 값이 유사하게 나타남을 알 수 있다. 반면에 블록 영역(Block Area)을 설정한 후 계산된 평균절대오차(MAE) 값은 침입자가 발생한 입력 영상 1인 경우 반복적인 움직임을 가지는 입력 영상 2나 입력 영상 3보다 상당히 커짐을 확인할 수 있다. 따라서 이렇게 블록 영역을 설정한 후 계산된 평균절대오차(MAE) 값을 시스템에서 설정한 임계값 30과 비교하여 임계값 30보다 크면 침입자가 발생한 영상이고, 그렇지 않으면 어떤 물체가 주변 환경에 의해 자연적으로 움직임, 반복적인 움직임의 발생한 영상으로 구분하여 인식 할 수 있다.

표 2. 평균 절대 오차

	전처리 과정 후		블록 영역 설정 후	
	이미지 크기 (N×M)	MAE	이미지 크기 (N×M)	MAE
입력 영상 1	320×240	6.1	52×142	62.1
입력 영상 2	320×240	6.2	137×144	22.0
입력 영상 3	320×240	8.5	75×128	21.8

4.5 배경 키 프레임 갱신

배경 키 프레임은 현재 입력 영상 키 프레임과 함께 평균절대오차(MAE)값을 구하는데 기준이 되는 중요한 요소이다. 만약 물체의 반복적인 움직임의 발생한 영상의 경우 배경 키 프레임을 갱신하지 않으면, 계산되는 평균절대오차(MAE)값은 점점 더 커지게 된다. 따라서 입력 영상 2, 3과 같은 물체의 반복적인 움직임이 발생한 영상인 경우에는 배경 키 프레임을 갱신해야 할 필요가 있다. 그림 9는 앞선 입력 영상 2의 선풍기의 회전처럼 바람에 의해 문이 열리는 반복적인 움직임이 발생한 영상을 가지고 배경 키 프레임을 갱신할 때와 그렇지 않을 때를 각각 보여주고 있다.

표 3은 그림 8 영상들에 대한 평균절대오차 값을 계산한 결과를 보여주고 있다.

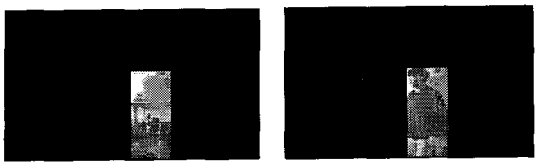
표 4를 살펴보면 배경 키 프레임의 현재 입력 영상의 직전 프레임으로 갱신되는 경우 그림 9에는 평균절대오차(MAE)값이 변화가 적음을 실험을 통해 확인할 수 있었다.

$$g_{b+1}(x,y)=\begin{cases} g(x,y) & (MAE<t) \\ g_{b+0}(x,y) & (MAE\geq t) \end{cases} \quad (10)$$

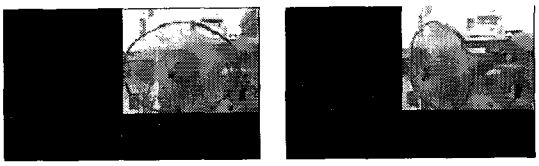
식 (10)에서 $g_{b+1}(x,y)$은 현재 갱신될 배경 키 프레임이고, $g_{b+0}(x,y)$는 이전의 배경 키 프레임, 그리고 $g(x,y)$는 현재 입력된 영상의 키 프레임, t는 임계값을 각각 나타내고 있다.

표 3. 평균절대오차 결과 값

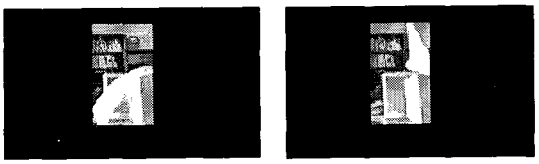
	1번째	2번째	3번째
배경 키 프레임의 갱신될 때 MAE : 그림 9	5.0	6.0	4.5



(a) 입력 영상 1



(b) 입력 영상 2



(c) 입력 영상 3

그림 8. 현재 영상에서 블록 영역 검출

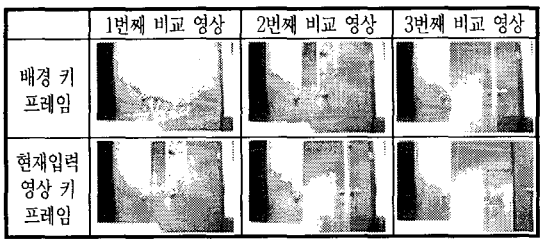


그림 9. 배경키 프레임 갱신

4.6 실험결과 분석

그림 10은 본 논문에서 구현한 전체 시스템의 모습을 보여주고 있으며 표 4는 본 논문에서 제시한 알고리즘을 앞서 설명한 3가지의 서로 다른 움직임의 발생한 입력 영상들을 가지고 각각의 모의 실험을 해본 결과를 차영상 움직임 검출 기법과 비교하여 보여주고 있다.

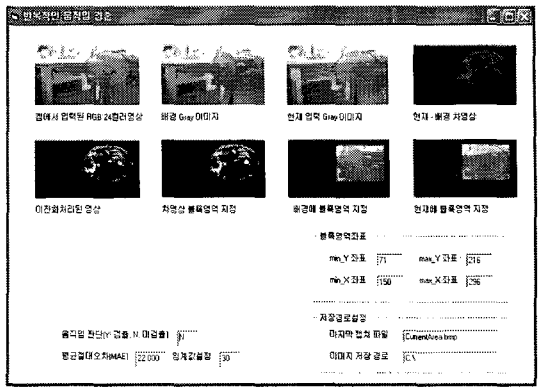


그림 10. 전체적인 시스템

표 4에서 평균 차 픽셀 계수는 배경 키 프레임과 현재 영상 키 프레임간 차영상에 의해 검출된 픽셀들의 계수이고, 평균절대오차 값은 배경 키 프레임과 현재 영상 키 프레임에 각각 블록 영역 좌표를 구하여 설정된 배경 영역과 현재 영역간 블록 정합시켜 계산된 평균절대오차 값을 나타낸다. 또한 검출오류는 현재 입력된 움직임의 발생한 총 영상 프레임의 계수를 움직임의 검출된 영상의 프레임 계수(F)로 나누어 백분율로 환산하여 계산된 값이다. 단, 입력 영상 1인 경우에는 움직임의 발생하지 않는 영상을 F값으로 설정하여 검출오류를 계산하였다. 그리고 계산된 평균처리시간은 계속적인 움직임의 발생한 60초 입력영상에서 움직임의 검출된 프레임의 개수를 가지고 평균처리 시간을 얻었다.

차영상 움직임 검출 기법인 경우에는 60초 동안 평균 309개의 움직임 영상을 얻었고, 본 논문에서 제안한 기법인 경우에는 평균 129개의 움직임 영상을 얻을 수 있었다. 표 5로부터 움직임 검출 오류를 측정한 결과 실험 입력 영상 1인 경우에 대해, 차영상 움직임 검출 기법에서는 6.2%, 제안한 알고리즘에서는 17.3%이며, 입력 영상 2에서는 각각 77.66%, 10.0%

표 4. 입력 영상에 대한 결과

분류 / 영상환경	차영상 기법				제안 알고리즘			
	평균 차 픽셀 계수	평균 MAE	검출오류 ((F/A)*100%)	평균처리 시간(sec)	평균 차 픽셀 계수	평균 MAE	검출오류 ((F/A)*100%)	평균처리 시간(sec)
입력영상 1	12002.6	3.8	6.2%	0.194	4282	67.6	17.3%	0.465
입력영상 2	20030	3.7	77.6%		2122	13.6	10.0%	
입력영상 3	12380	9.8	91.2%		8524	16.5	13.3%	

이고 입력 영상 3에서는 91.2%, 13.3%로 나타났으며 처리속도는 차영상 움직임 검출 기법이 제안한 알고리즘 보다 약 2.4배 빠르게 검출됨을 확인 할 수 있었다. 또한, 실험 입력 영상 2나 입력 영상 3의 환경처럼 물체의 반복적인 움직임이 있는 영상에서는 제안한 움직임 검출 기법이 차영상 움직임 검출 기법보다 상당히 좋은 결과를 얻을 수 있었으나 입력 영상 1처럼 침입자가 발생한 영상인 경우에는 좋지 않은 결과를 보였다.

5. 결 론

감시 시스템에서 움직임 검출 기법은 상당히 중요한 비중을 차지하는 부분으로 얼마나 정확하게 움직임의 발생하였는지 검출하여 저장하는 것은 감시 시스템의 성능을 좌우하는 중요한 요소들 중 하나이다.

이러한 감시 시스템에서 움직임을 검출하는 방법에는 가장 빠른 처리 속도를 가지는 차영상 움직임 검출 기법(The Motion Detection using Difference Image)이 가장 많이 사용되고 있으나, 이 움직임 검출 기법은 배경의 명암도의 변화나 잡음 및 반복 적인 물체의 움직임에 대해서는 많은 문제점이 발생한다. 본 논문에서는 이렇게 차영상 움직임 검출 기법에서 문제점으로 대두되고 있는 반복적인 움직임의 발생하였을 경우에 대해 그 해결 방법을 제안하고 있다. 차영상을 통해 움직임의 검출 되었을 때, 이 움직임의 물체의 반복적인 영상인지, 아니면 침입자가 발생한 영상인지를 구분하기 위해 본 논문에서는 물체의 반복적인 움직임인 경우에 현재 입력 영상 키 프레임(Current Key Frame)과 배경 영상 키 프레임(Background Key Frame)과의 움직임의 검출된 영역 간 픽셀 값의 분포도가 매우 차이가 크다는 특징을 이용하였다. 특징을 적용하기 위해 먼저 현재 입력 영상 키 프레임과 배경 영상 키 프레임과의 차영상을 구하고, 블록 영역의 좌표를 구하기 위해 차영상을 이진화 처리하여 움직임의 검출된 영역만을 블록 영역(Block Area)의 좌표로 지정하였다. 이렇게 지정된 블록 영역 좌표를 현재 입력 영상 키 프레임과 배경 영상 키 프레임에 대입하여 현재 블록과 배경 블록을 얻고, 이 두 블록을 정합시켜 평균절대오차(MAE) 값을 계산한다. 계산된 평균절대오차 값이 시스템에서 설정된 임계값 보다 크면 침입자가 발생한 것으로 반대로 작으면 물체가 주변 환경에 의해 자동으로 반복적인 움직임 영상으로 인식되도록 하였다. 실험 결과 본 논문에서 제시한 방법으로 움직임을 검출 하였을 때 물체의 반복적인 움직임의 발생한 영상에서는 차영상 움직임 검출 기법과 비교하여 약 77%의 성능 개선을 보였다. 향후 본 논문을 좀더 개선하여 침입자가 발생한 영상인 경우에 있어서 움직임 검출 처리 속도와 성능을 향상시킨다면 무인 감시 시스템에 적용하여 감시 시스템의 성능을 향상 시킬 수 있는 결과를 제시하고자 한다.

참 고 문 헌

[1] N. McFarlance, "Segmentation and Tracking of Piglets in Image", Machine Vision Application, Vol. 8, pp187-193, 1995.

[2] C. Wren, A. zarbayejani, T. Darrell and A. Pentland, "Pfinder : Real-Time Tracking of the Human Body", IEEE Trans. Pattern Analysis and Machine Intelligence, Vol. 9, No. 7, 1997.

[3] Y. Ivanov, A. Bovick, and J. Liu, "Fast Lighting Independent Background Subtraction", Technical Report, No. 437, MIT Media Lab, 1997.

[4] Willian B. Thompson, Pamela Lechleider and Elizabeth R. Stuck, "Detecting moving objects using the rigidity constranint", IEEE Transaction on Pattern Analysis and Machine Intelligence, PAMI-15, No. 2, pp. 162-169, 1993.

[5] Vincent S. S Hwang, "Tracking Feature points in Time-Varying images using an opportunitic selection approach," Pattern Recognition Vol. 22, No. 3, pp. 247-256, 1999.

[6] 이규원, 김영호, 이재규, 박규태, "무인 감시 장치 구현을 위한 단일 이동물체 추적 알고리즘", 전자공학회 논문지, 31권 B편, 11호, 1995.

[7] 하영현, 채옥삼, "차영상의 차 히스토그램을 이용한 자동 임계값 결정", 신호처리 종합 학술대회 논문집, 11권, 1호, 1998.

[8] 조태훈, 최영규, "다중 배경 분포를 이용한 움직임 검출", 한국정보처리학회 논문집, 8권, 5호, 2001.

[9] 조영식, 이주신, "부분 외곽선 정보를 이용한 이동물체의 추적 알고리즘", 정보처리학회 논문지, 8권, 5호, 2001.

[10] 조영창, 이태홍, "움직임 영역간 보상오차의 최소편차를 이용한 최적 블록정합 움직임 추정", 정보처리학회 논문지, 8권, 5호, 2001.

강 진 석

1999년 제주대학교 정보공학과(공학사)
2001년 2월 제주대학교 대학원 정보공학과(공학석사)
2001년 2월~현재 제주대학교 정보공학과 대학원 박사수료
2004년 9월~현재 군산대학교 BK산학력팀 계약교수
관심분야 : 멀티미디어 시스템, 영상처리

김 장 형

1981년 2월 홍익대학교 정밀기계공학과(공학사)
1983년 2월 연세대학교 대학원 기계공학과(공학석사)
1990년 8월 홍익대학교 대학원 기계공학과(공학박사)
1998년 3월~2000년 5월 제주대학교 전자계산소장
1984년 2월~현재 제주대학교 통신컴퓨터공학부 교수
관심분야 : CAD/CAM, 멀티미디어, 인공지능