creative commons

COMMONS DEED

저작자표시-비영리-변경금지 2.0 대한민국

이용자는 아래의 조건을 따르는 경우에 한하여 자유롭게

- 이 저작물을 복제, 배포, 전송, 전시, 공연 및 방송할 수 있습니다.

다음과 같은 조건을 따라야 합니다:

저작자표시. 귀하는 원저작자를 표시하여야 합니다.

비영리. 귀하는 이 저작물을 영리 목적으로 이용할 수 없습니다.

변경금지. 귀하는 이 저작물을 개작, 변형 또는 가공할 수 없습니다.

- 귀하는, 이 저작물의 재이용이나 배포의 경우, 이 저작물에 적용된 이용허락조건을 명확하게 나타내어야 합니다.
- 저작권자로부터 별도의 허가를 받으면 이러한 조건들은 적용되지 않습니다.

저작권법에 따른 이용자의 권리는 위의 내용에 의하여 영향을 받지 않습니다.

이것은 이용허락규약(Legal Code)을 이해하기 쉽게 요약한 것입니다.

Disclaimer

dCollection

[UCI]I804:24011-200000264247

2014年 2月

석사학위논문

누적 움직임벡터를 이용한 겹침현상에 강인한 보행자 추적

조 선 대 학 교 대 학 원

정보통신공학과

김 정 묵

2014년 2월 석사학위논문

누적 움직임벡터를 이용한 겹침현상에 강인한 보행자 추적

김 정 묵

누적 움직임벡터를 이용한 겹침현상에 강인한 보행자 추적

A Robust Pedestrian Tracking in the Presence of Occlusion Using Accumulated Motion Vector

2014년 2월 25일

조 선 대 학 교 대 학 원

정보통신공학과

김 정 묵

누적 움직임벡터를 이용한 겹침현상에 강인한 보행자 추적

지도교수 권 구 락

이 논문을 공학석사학위신청 논문으로 제출함

2013년 10월

조 선 대 학 교 대 학 원

정보통신공학과

김 정 묵

김정묵의 석사학위 논문을 인준함

위원장 조선대학교 교수 박 종 안 印

위 원 조선대학교 교수 신 영 숙 印

위 원 조선대학교 교수 권 구 락 印

2013년 11월

조 선 대 학 교 대 학 원

목 차

그 림 목 차 ····· ii
표 목 차 ····· iii
ABSTRACT ····· iv

Ⅰ. 서 론 ····· 1
A. 연구 배경 및 목적 ····· 1
B. 연구내용 ····· 3
Ⅱ. 보행자 특징 추출 및 학습과정 ····· 4
A. 보행자 특징 추출 방법 ····· 4
1. 블록 정합 알고리즘 ····· 5
2. Haar-like 특징 ····· 6
3. HOG(Histogram of Oriented Gradient) ····· 9
B. 보행자 특징 학습 과정 ····· 13
1. SVM(Support Machine Vector) ····· 14
2. Adaboost알고리즘 ····· 15
C. Dalal이 제안한 보행자 추적 알고리즘 ····· 17
Ⅲ. 제안하는 보행자 추적 알고리즘 ····· 20
Ⅳ. 실험 및 고찰 ····· 23
A. 실험 순서 ····· 23
B. 제안하는 방법의 보행자 추적 실험 ····· 27
Ⅴ. 결론 및 향후 연구과제 ····· 39
참고문헌 ····· 40

그 림 목 차

그림 2.1 기본적인 객체 인식 과정 ····· 4
그림 2.2 블록 정합 알고리즘 ····· 5
그림 2.3 Haar-like 특징 마스크 ····· 7
그림 2.4 적분 이미지 ····· 7
그림 2.5 적분 이미지를 이용한 계산 ····· 8
그림 2.6 HOG특징값 생성 및 기울기 특징의 시각화 ····· 9
그림 2.7 HOG를 이용한 셀 정규화 과정 ····· 11
그림 2.8 보행자의 특징값 파라미터 학습 과정 ····· 13
그림 2.9 SVM분류기의 학습 방법 ····· 13
그림 2.10 Adaboost 학습 방법 ····· 15
그림 2.11 일반적인 보행자 추적 순서 ····· 17
그림 2.12 Dalal이 제안한 HOG/SVM을 이용한 특징학습 및 보행자 분류 과정 ····· 18
그림 2.13 보행자의 일부분 겹침현상의 개선 ····· 19
그림 3.1 제안하는 보행자 추적 알고리즘 ····· 20
그림 3.2 HOG/Adaboost를 이용한 보행자 특징학습 및 보행자 분류 과정 ····· 21
그림 3.3 설정된 보행자 영역의 예 ····· 21
그림 3.4 움직임벡터 생성 및 예측영역 탐색의 예 ····· 22
그림 4.1 실험에 사용된 보행자 사진 데이터 ····· 24
그림 4.2 실험에 사용된 비보행자 사진 데이터 ····· 25
그림 4.3 Dalal의 방법과 제안하는 방법의 보행자 검출 결과 비교 ····· 27
그림 4.4 Dalal의 방법과 제안하는 누적 움직임벡터를 이용한 보행자 검출 결과 비교 ····· 28
그림 4.5 실험에 사용된 겹침현상을 가정한 프레임단위 영상(데이터 A) ····· 29
그림 4.6 실험에 사용된 겹침현상을 가정한 프레임단위 영상(데이터 B) ····· 30
그림 4.7 데이터 A 영상에서의 보행자 검출 및 추적 결과 ····· 32
그림 4.8 데이터 B 영상에서의 보행자 검출 및 추적 결과 ····· 34
그림 4.9 원본영상과 검출영상의 보행자 의상 색상 및 위치 비교 ····· 36
그림 4.10 데이터 A의 기존 보행자(a)와 예측 영역의 보행자(b)의 RGB색상 분포 곡선 ····· 37
그림 4.11 데이터 B의 기존 보행자(a)와 예측 영역의 보행자(b)의 RGB 색상 분포 곡선 ····· 38

표 목 차

표 4.1 실험에 사용된 시스템 사양 및 시뮬레이션 환경 ·····23
표 4.2 데이터 A의 (a)~(c) 프레임 움직임벡터와 예측 이동 방향 ·····33
표 4.3 데이터 B의 (a)~(d) 프레임 움직임벡터와 예측 이동 방향 ·····35
표 4.4 Dalal의 방법과 제안하는 방법의 보행자 검출 비교표(단위 : %) ·····36

ABSTRACT

A Robust Pedestrian Tracking in the Presence of Occlusion Using Accumulated Motion Vector

Kim, Jeong-Muk
Advisor : Prof. Kwon, Goo-Rak, Ph.D.
Department of Information Communication Engineering
Graduate School of Chosun University

In this paper, a pedestrian tracking method that meets the surveillance system using the gradient histogram and cumulative motion vector is proposed. In order to get the feature of pedestrian, a block-by-block histogram is created using the direction of the gradient based on HOG (Histogram of Oriented Gradient), and then only pedestrians are classified by learning the Adaboost in the video. Next, the motion vector is cumulated based on the area of pedestrian detection. In the case of overlap, the direction of pedestrian using the obtained cumulative motion vector is predicted. Through experiments, it is proven that the proposed method is more effective compared to the existing methods for the accurate tracking of the pedestrian.

I. 서 론

A. 연구 배경 및 목적

CCTV 영상은 많은 정보를 내포하고 있으며 이 정보들을 가공하고 처리하여 범죄예방과 수사에 큰 역할을 하고 있다. 전 세계적으로는 약 3,400만대가 설치되고 운영되고 있는 것으로 추산된다 [1]. 과거 CCTV는 단순히 일정공간을 촬영하여 특정장소에서 확인하는 역할만 하는 단순한 기능만을 가지고 있었다. 사회가 복잡해짐에 따라 강력범죄가 증가하고 교통량이 복잡해져 요구사항은 다양해졌다. CCTV설치는 해마다 증가하고 있지만, 아직 대부분의 시스템이 단순한 영상저장의 한계를 벗어나지 못하고 있다. 이를 위해 CCTV 기반 지능형 영상보안 기술에 대한 연구가 활발히 진행되고 있다. 카메라로부터 얻어진 영상을 이용해 움직이는 객체를 검출하고 추적하기 위한 연구는 활발히 진행중이다 [2]. 움직이는 객체는 주로 사람, 자동차, 동물 등 다양하나 대부분 사람, 차량, 얼굴 등을 검출하고 추적하는 연구가 주로 진행된다 [3]-[5].

현재 CCTV 시스템은 지능형 감시 시스템(Intelligent Surveillance System)으로 변화하고 있다. 지능형 감시 시스템은 동적 이동체의 검출, 분류, 인식, 추적을 위한 강인하고 신속하며 신뢰성 있는 알고리즘을 요구한다 [10][11]. 일반적으로 보행자를 검출하기 위해서는 하나의 카메라를 통해 얻어진 단일 영상 획득 방법과, 두 대 이상의 카메라를 통해 얻은 스테레오 영상의 깊이(Depth)를 이용하는 방법이 있다. 그리고 보행자를 한번씩 검출하는 방법과 보행자가 여러 그룹으로 있을 때 검출하는 방법, 보행자의 외형과 움직임벡터로 검출하는 방법이 있다 [12]-[15]. 하지만 하나의 영상을 이용하는 방법은 외부 환경에서의 빛과 배경, 보행자의 움직임과 자세와 같은 상태변화에 영향을 받는다.

보행자 검출을 위한 특징 추출 방법은 Haar-like특징(Haar-like feature), HOG(Histogram of Oriented Gradient)를 이용한 방법 등이 사용되며 이를 이용한 분류기 생성 학습에는 SVM(Support Vector Machine)와 Adaboost(Adaptive boosting) 그리고 신경망 분류 알고리즘이 많이 적용되고 있다 [6]-[9].

Haar-like 특징은 Viola와 Jones에 의해 제안된 방법으로 4종류의 사각형을 이용한 적분 영상을 통해 빠른 속도로 계산하여 얼굴 및 보행자 검출 등에 적용된 방법이다.

좋은 성능을 보이긴 하나 주변 환경의 변화와 같은 사각형이 인접한 형태로는 보행자의 다양한 특징을 얻는데 물리적 구조의 한계를 보여준다 [6].

Dalal이 제안한 HOG 특징은 객체 크기와 조명의 변화 등 주변 환경의 변화에서도 객체의 특징을 잘 표현한다. HOG는 기울어짐이나 크기의 변화 등의 왜곡에 대해서 보다 안정적으로 보행자의 특징을 나타낼 수 있다 [7].

SVM은 이진 분류기로서 특징 벡터를 이용해 학습하고 1:1 매칭 기법을 이용해서 식별하는 역할을 수행한다. 기본적으로 1:1 매칭이기 때문에 미리 구축된 데이터베이스와 연동하여 사용하는 것이 일반적이다 [8]. 실시간 상황을 바탕으로 하는 경우에는 SVM보다는 학습을 통해 강한 분류기를 만드는 Adaboost를 사용한다.

Adaboost는 연산량이 적은 다수의 약한 분류기(Weak Classifier)중에서 성능별로 선택하여 강한 분류기(Strong Classifier)를 구성한다. 이 강한 분류기는 전체 연산량이 적고, 검출속도가 우수하며, 검출 능력도 우수해 현재 많은 응용 분야에서 사용되어지고 있다 [9].

그러나 아무리 좋은 성능의 특징 추출과 분류기를 통해서 보행자를 검출한다 해도, 추적하고 있는 보행자가 다른 배경요소에 의해서 일부분이 사라지거나 전체 영역이 가려질 경우에는 여전히 추적에 실패한다. 그리고 위와 같은 상태에서 벗어나 다시 보행자가 나타난다 해도 기존에 추적하고 있는 보행자인지 아닌지는 판단 할 수 없다. 이러한 문제점을 개선하고자 아래와 같은 방법으로 연구를 진행하였다.

B. 연구 내용

본 논문에서는 보행자 검출 및 추적을 보행자의 특징을 학습하여 분류기를 생성하고 검출 한 후 추적한다 [16][17]. 카메라 영상에서 보행자의 특징을 HOG(Histogram of Oriented Gradient)를 이용하여 검출 및 추적한다. HOG는 보행자 특징 추출에 있어서 Haar-like 특징보다 확실한 값을 얻는다. 보행자의 전신 영역의 영상에서 기울기의 방향성 특징 벡터를 생성하고 Adaboost 알고리즘 학습을 통해 보행자만을 분류한다. 이후 검출된 보행자의 위치 정보를 이용하여 보행자 추적을 실시한다. 또한 다른 객체와의 겹침이 발생할 때 추적에 실패하는 경우를 해결하기 위해 추적하고자 하는 보행자를 중심으로 블록 정합 알고리즘을 이용해 움직임 벡터를 얻는다 [18]. 영상 전체를 블록 매칭 알고리즘을 이용하면 처리시간의 문제가 생긴다 [19]. 이를 보완하기 위해 추적하고 있는 보행자 주변의 일정 영역을 관심 영역으로 설정하고 해당 영역에서만 움직임 벡터를 얻어내어 처리시간을 단축한다. 얻어낸 움직임 벡터값은 프레임별로 누적시켜 평균 움직임 벡터를 얻어내어 겹침이 일어나더라도 관심 영역은 추적하려는 대상이 나타날 수 있는 예상 지점으로 이동된다. 이렇게 이동된 관심영역은 기존의 보행자를 놓치지 않고 추적할 수 있다. 본 논문의 구성은 다음과 같다. 본 논문의 2장에서는 일반적인 보행자 인식 및 추적 방법과 기존의 보행자 추적 방법에 대해서 설명하고, 3장에서는 제안하는 알고리즘에 대해서 설명한다. 4장에서는 실험 및 고찰을 통해 성능 비교를 하며, 5장에서는 연구의 결론 및 향후의 연구 방향을 기술한다.

II. 보행자 특징 추출 및 학습 과정

A. 보행자 특징 추출 방법

비전 처리 분야에서 객체 인식은 보행자와 차량 및 사람의 얼굴에 대한 데이터베이스를 학습하여 얻어진 특징들을 이용해 인식하게 된다. 그림 2.1과 같은 순서로 임의의 객체가 입력되었을 때 이 객체와 기존에 추출된 특징을 비교해 유사하다고 판단되었을 경우 인식하여 찾아내거나 추적하는 과정을 거친다 [10].

그림 2.1 기본적인 객체 인식 과정

보행자 인식은 보행자만이 가지고 있는 특징값을 이용하여 판별하는 것이다. 이러한

보행자 인식의 어려운 점은 같은 대상이라 하더라도 입력되는 영상에서는 방향 또는 크기에 따라 인식에 실패하는 경우와 한 영상에서 어려개의 객체가 포함되는 배경이 복잡한 경우, 인식하려는 보행자가 다른 물체에 가려져 일부만 나타나는 경우 등이 있다.

보행자를 정확하게 인식하기 위해서는 대상에 대한 다양한 형태의 데이터가 필요하다. 이러한 데이터들을 통하여 인식 대상의 특징 모델을 설정하게 된다. 특징을 이용하면 입력되는 영상에서 찾고자 하는 대상이 있는지 판별할 수 있는 파라미터로 이용할 수 있다. 이를 위해서는 데이터를 이용한 특징 학습 과정이 필요하다. 아래는 특징 추출 방법에 대해서 설명한다.

1. 블록 정합 알고리즘

본 논문에서는 얻어낸 움직임벡터를 이용하여 겹침현상에서 보행자의 움직임을 예측하는 용도로 사용한다.

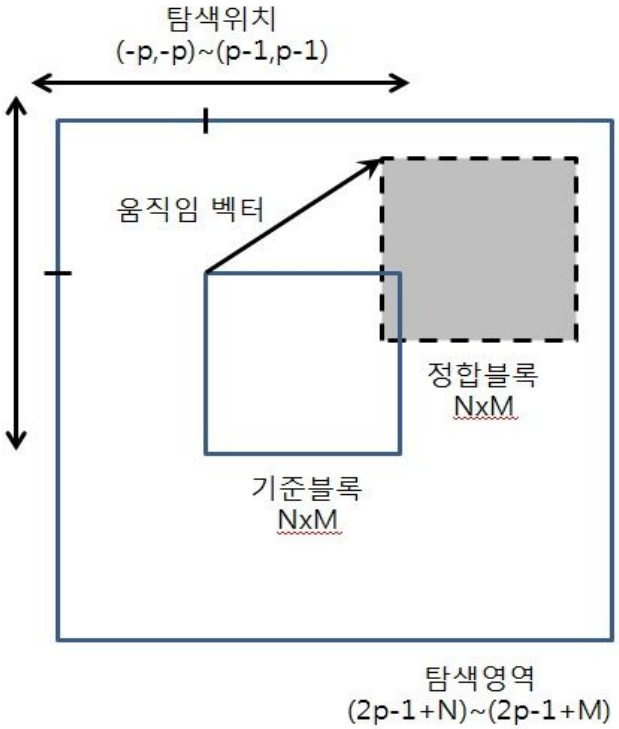


그림 2.2 블록 정합 알고리즘

블록정합알고리즘(BMA: Block Matching Algorithm)은 영상처리과정에서 지역적 움직임 특징을 얻어내는 기본적인 알고리즘이다 [19]. 이를 이용하여 보행자의 움직임 방향을 얻어낼 수 있다.

첫번째로 현재프레임을 N×M인 작은 블록으로 나눈 후 참조 프레임에서 현재 프레임 블록에 대응되는 블록보다 큰 영역을 취하여 탐색 창을 정한다. 이 탐색 창 내에서 현재프레임 블록과 가장 정합되는 블록을 찾게 되면 이 블록의 모션벡터를 얻게 된다.

매칭블록은 식(1), 식(2)와 같은 정합기준(Matching Criterion)을 이용하며 오차가 제일 작은 블록이 가장 잘 정합되는 블록이라고 볼 수 있다.

$$MSD(f) = \frac{1}{MN}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}\left[f_k(m,n) - f_{k-N}(m+i,n+j)\right]^2 \qquad (1)$$

$$MAD(f) = \frac{1}{MN}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}\left|f_k(m,n) - f_{k-N}(m+i,n+j)\right| \qquad (2)$$

식(1)은 평균 제곱 오차(MSD; Mean Squared Difference) 함수이고 식(2)는 평균 절대 오차(MAD; Mean Absolute Difference) 함수이다. 실제 응용에서는 식(3)과 같이 절대 값의 합(SAD; Sum of Absolute Difference)로서 MAD를 대체하기도 한다.

$$SAD(u,v) = \sum_{i=1}^{N}\sum_{j=1}^{M}\left|R(i,j) - S(i+u,j+v)\right|, \qquad -p \le u,v < p \qquad (3)$$

BMA은 정확성이 높지만(특히 Full Search) 계산 양이 많아 연산 속도가 느리다는 단점을 갖고 있다. 그러나 본 논문에서는 영상 전 영역에 대해 움직임 벡터를 추출하는 것이 아닌 보행자와 주변의 일정 영역을 관심영역으로 하여 해당되는 부분에 대한 움직임 벡터를 추출한다. 추출한 움직임 벡터는 프레임별로 합산 한 후 평균을 구해 사용한다. 이렇게 하면 보행자의 평균적 이동 거리를 구할 수 있다.

2. Haar-like 특징

Haar-like 특징은 Viola가 제안한 방법으로 적분 영상을 통해서 빠른 속도로 계산하여 특징을 얻는다. 얼굴 검출에 적용된 방법이었으나, 보행자 검출에도 사용되었다. 하

나의 검은색 사각형 위치와 가로, 세로 크기가 정해진 후 이와 같은 크기의 흰색 사각형을 가로, 세로, 대각선으로 붙여서 두 개의 면적을 구하여 이용하는 방법이다 [6].

(a) (b) (c) (d)

그림 2.3 Haar-like 특징 마스크

그림2.3의 (a)는 수직, (b)는 수평, (c)는 좀 더 넓은 수직 성분 (d)는 대각선 방향의 특징 성분을 추출하는데 사용된다.

Haar-like는 직사각형의 형태이므로 적분 이미지를 이용해 빠른 속도로 계산이 가능하다. 점(x,y)에서 적분 이미지 값은 그림 2.4와 같이 왼쪽 위 끝점에서부터 점(x,y)까지 픽셀들의 합이다.

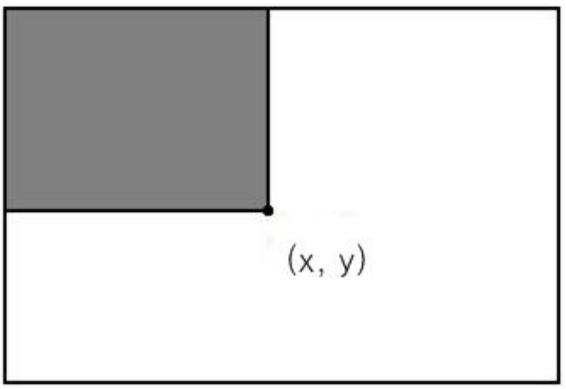


그림 2.4 적분 이미지

적분 이미지 ii(x,y)는 식 (4)와 같이 표현할 수 있다. 여기서 i(x,y)는 원본 이미지이다.

$$ii(x,y) = \sum_{x' \le x, y' \le y} i(x',y') \tag{4}$$

여기서 식(4)는 식(5)와 (6)을 반복적으로 사용하여 얻는다.

$$s(x,y) = s(x,y-1) + i(x,y) \tag{5}$$

$$ii(x,y) = ii(x-1,y) + s(x,y) \tag{6}$$

여기서 $s(x,y)$: 열의 누적합

$s(x,-1), ii(-1,y) = 0$

적분 이미지는 처음 영상이 들어왔을 때, 픽셀값을 합한 한 번의 계산으로 SAT(Summed Area Table)을 생성한다. 이렇게 구해진 적분 이미지를 이용하면 실제 원하는 위치의 픽셀값들의 합을 구할 때 빠른 계산이 가능하다. 그림 2.5의 D영역의 픽셀값의 합을 구하는 과정은 다음과 같다. 점1에서의 적분 이미지값을 A영역의 각 픽셀값을 모두 합한 값이라고 할 수 있다. 점2에서의 적분값은 A영역과 B 영역 픽셀값의 합이다. 마찬가지로 점3에서의 적분 이미지의 값은 A영역과 C영역의 픽셀값의 합이고 점4에서의 적분 이미지값은 모든 영역의 픽셀값의 합이다.

각 점의 적분 이미지값을 1부터 4라고 할 때 D영역의 픽셀값의 합은 4+1 - (2+3)을 계산하여 얻을 수 있다. 위의 방법을 통해 하나의 직사각형 특징값을 4개의 적분 이미지값의 덧셈과 뺄셈으로 얻을 수 있다. Haar-like 특징은 이웃한 직사각형으로 이루어져 있으므로 두 개 의 직사각형으로 이루어진 특징은 여섯 점의 적분 이미지 값을 이용하여 구할 수 있다.

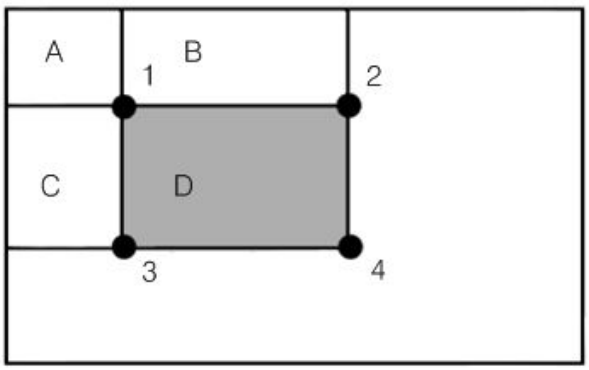


그림 2.5 적분 이미지를 이용한 계산

Haar-like 는 간단하게 에지(Edge)를 이용한다고 할 수 있다. 그러나 에지가 주위 이웃한 것에 대해서 분류한다고 하면 Haar-like는 영역을 기반으로 하기 때문에 값의 분포 범위가 넓고 이진 분류의 기준이 모호하다. 이것은 좋게 보면 잡음에 둔감하다는 뜻이지만, 잡음에 둔감하다는 것은 나쁘게 보자면 하나의 특징으로는 조금 부족하다고 볼 수 있다.

3. HOG(Histogram of Oriented Gradient)

HOG는 영상을 지역 히스토그램의 집합으로 표현한다. 보행자 또는 사람의 얼굴과 같은 다양한 패턴을 갖는 객체를 인식하기에 적합한 특징을 추출하는 방법이다. 기본적인 알고리즘의 개념은 일정 셀을 블록화하여 블록의 밝기 값 및 방향값의 히스토그램을 연접하여 특징 값을 추출하는 것이다 [7].

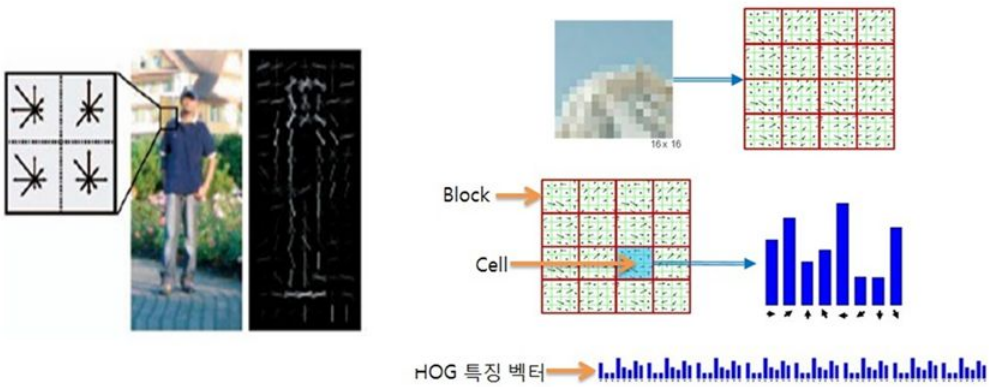


그림 2.6 HOG특징값 생성 및 기울기 특징의 시각화

HOG 특징의 기울어짐, 배경과 객체간의 대비와 잡음에 강인한 특징을 이용해 보행자의 분류가 가능하다. HOG는 영상내의 지역 영역의 기울기 방향 정보의 발생 빈도를 누적시킨 후 히스토그램을 생성한다. 이 방법은 근접화소의 분포를 국소 영역에 대해서 히스토그램화하며, 주변 밝기의 영향에 강인하며 기하학적 변화에 강인한 특징을 가진다. 일반적으로 HOG의 특징 벡터를 구하는 과정은 세 단계를 거친다.

첫 번째로 영역별 기울기 값을 계산한다. 두 번째로 셀 단위의 누적 히스토그램을 구하며 마지막으로 블록 내의 셀을 정규화한다.

기울기 값은 각 픽셀의 밝기 영상으로부터 x축과 y축의 변화에 대한 분포의 크기 m

과 밝기영상 i(x,y)의 변화에 대한 방향성을 나타내는 θ를 다음 식(6), (7), (8)을 이용하여 각각 계산한다.

$$m(x,y) = \sqrt{f_x(x,y)^2 + f_y(x,y)^2} \tag{6}$$

우선 m은 밝기영상 i(x,y)로부터의 x축과 y축 변화에 대한 분포의 크기를 나타낸다.

$$\theta(x,y) = \tan^{-1}\frac{f_y(x,y)}{f_x(x,y)} \tag{7}$$

θ는 밝기영상 i(x,y)의 변화에 대한 방향성을 나타낸다.

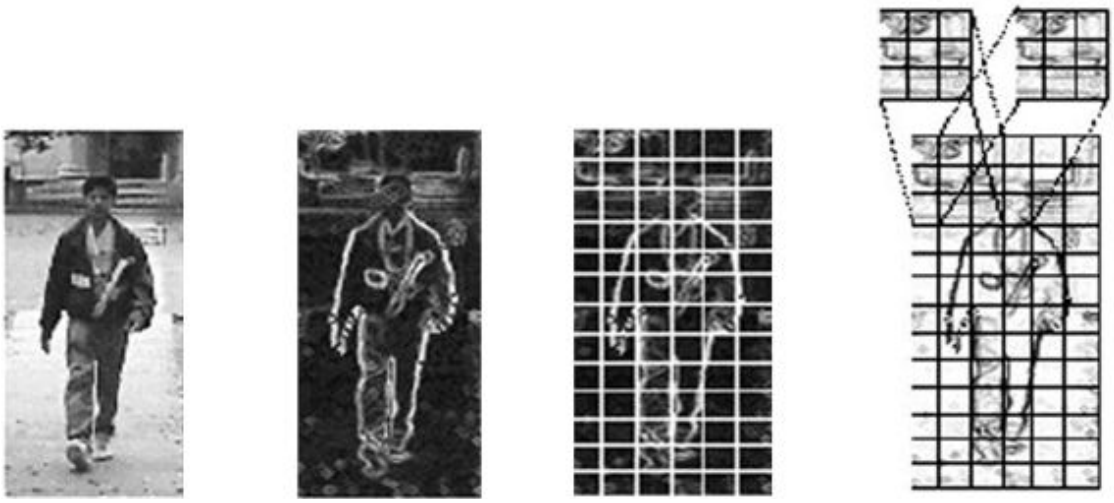
$$\begin{cases} f_x(x,y) = i(x+1,y) - i(x-1,y) \\ f_y(x,y) = i(x,y+1) - i(x,y-1) \end{cases} \tag{8}$$

$f_x(x,y)$와 $f_y(x,y)$는 밝기 영상에 대한 밝기의 분포를 나타낸다.

기울기에 의해서 얻어진 방향은 0에서 2π의 범위를 가진다. 이 값은 연산이 가능하도록 몇 개의 값으로 양자화되며 히스토그램이 갖게 되는 값의 개수를 결정한다.

계산된 변화에 대한 밝기영상분포의 크기를 나타내는 m과 변화의 방향성을 나타내는 θ를 이용해서 보통 8x8 크기의 셀(Cell)을 설정해 사용한다. 이 셀의 밝기 변화에 관한 히스토그램을 작성한다.

HOG는 영상을 여러 셀로 나누는 특이성이 있다. 픽셀단위 크기가 정해지면 셀 단위로 각 방향의 정해진 히스토그램을 구하기 위해서 방향에 따라 기울기의 크기 즉, 에지성분을 누적시킨다. 이를 통해서 에지가 강한 영역과 그렇지 않은 영역의 구분이 가능하게 된다. 모든 셀에 히스토그램이 계산되면 얻어진 기울기 값의 누적된 히스토그램은 하나의 벡터로서 연결이 가능하게 된다. 이는 분류기(Classifier)입력에 적합하게 된다.



(a) (b) (c) (d)

그림 2.7 HOG를 이용한 셀 정규화 과정

그림 2.7는 셀 표현의 예를 나타낸다. (a)는 입력된 영상이며 (b)는 HOG 크기 영상이다. (c)는 HOG 크기영상을 셀 단위로 분류한 것이며 이때 하나의 셀은 8X8의 픽셀로 구성된다. (d)는 3x3의 셀 집합을 하나의 블록으로 사용한다.

기울기의 계산, 셀과 블록 히스토그램 과정을 통해 얻어진 벡터는 부분적인 조명 차이, 주야간 밝기, 반사광 등의 각종 영상 잡음의 영향을 피해야 한다. 이는 각 셀의 히스토그램 정규화 과정이 필요함을 의미한다. 셀 히스토그램은 인접한 셀의 히스토그램 값을 이용하여 지역적으로 정규화된다. 정규화는 블록(Block)이라 불리는 여러 셀을 그룹으로 한다.

각 셀에서 얻어진 밝기에 대한 방향성 히스토그램은 3x3셀을 하나의 블록으로 하는 정규화를 실시한다. I행 j열의 셀의 특징량(9차원 기준)은 $F_{i,j} = [f_1, f_2, ...f_9]$ 로 나타낸다. k번째 블록의 특징량(81차원 기준)은

$B_k = [F_{i,j}, F_{i+1,j}, F_{i+2,j}, F_{i,j+1}, F_{i+1,j+1}, F_{i+2,j+1}, F_{i,j+2}, F_{i+1,j+2}, F_{i+2,j+2}]$로 표시할 수 있다. 이는 그림 2.5와 같이 HOG특징벡터를 하나로 연결한 것과 같다. 이렇듯 정규화를 마친 모든 셀의 히스토그램은 하나의 벡터로 연결된다. 이렇게 얻어진 특징벡터를 V라고 하면 정규화 방법은 식(7)과 같다.

$$v = \frac{f}{\sqrt{\| B_k \|_2^2 + \epsilon^2}}, (\epsilon = 1) \tag{7}$$

정규화 과정을 그림 2.6과 같이 블록의 이동은 오른쪽과 아래로 1셀씩 이동는 것을 기준으로 하였다. 입력 영상이 64x128픽셀의 경우 가로로 6블록, 세로로14블록이 생성된다.

ϵ은 기울기 값이 없는 경우 연산을 위해 필요하며 매우 적은 값을 가지므로 인식 결과에 영향을 미치지 않는다. 셀과 블록의 크기가 크면 연산량은 감소하나 원본 영상의 정보가 손실될 수 있다. 그러므로 인식 성능에 문제가 없는 범위에서 큰 값을 취하는 것이 적절하다. 이를 통해 HOG 특징을 적용하는 대상에 따라 최적의 결과를 얻는 것이 가능하다.

B. 보행자 특징 학습 과정

학습 과정은 인식 대상의 특징 파라미터를 생성하는 것으로 인식 대상의 영상과 인식의 대상이 아닌 배경 영상을 이용하여 보행자인지 비보행자인지를 판단한다. 이렇게 얻어진 특징값을 이용하여 입력 영상에서 보행자가 있는지 없는지를 판별하게 된다 [8][9].

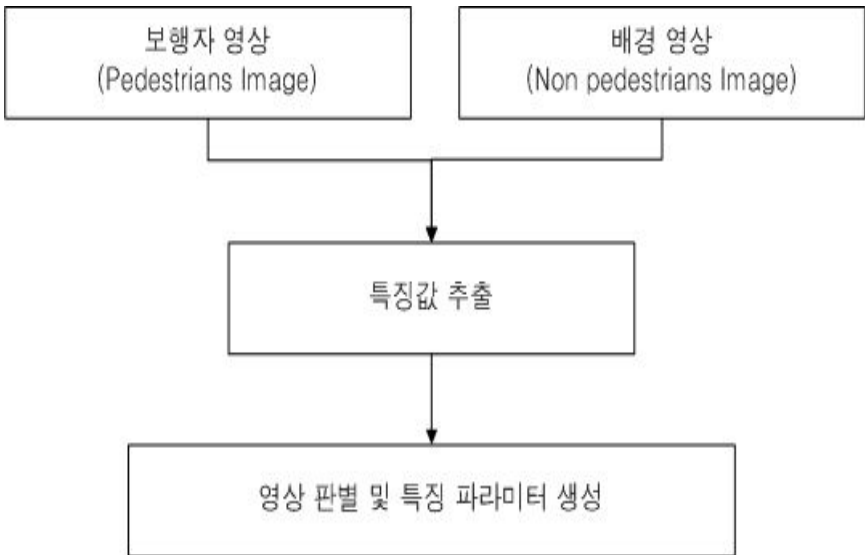


그림 2.8 보행자의 특징값 파라미터 학습 과정

보통 특징값을 추출하고 인식한 후엔 아래와 같은 순서로 보행자를 판별하고 추적하게 된다. 위의 그림에서 특징 파라미터를 얻기 위해 주로 사용되는 보행자 특징 학습 방법은 주로 2가지가 사용된다. SVM과 Adaboost 방법이 대표적인 특징 학습 알고리즘이다. 각 방법의 특징은 다음과 같다.

1. SVM(Support Vector Machine)

SVM은 신경망 등 다른 분류기와는 학습 과정이 다르다. 신경망 방법은 학습을 해나가면서 최소 오류 지점에 다다르면 학습을 종료한다. 이것은 학습 데이터에 대해서 오류를 최소화하여 가장 오류율이 적은 하나의 분류기를 만드는 것에 의의를 갖는다. 하지만 SVM은 오류율보다 분류해야 하는 두 부류의 여백을 극대화하는 것에 초점이 맞춰진 분류기이다 [8].

SVM 분류기는 기본적으로 이진 분류기이기 때문에 그 특징을 이용하여 각 부류에 SVM 분류기를 두고 입력 데이터에 대한 각 부류의 1:1 매칭을 한다.

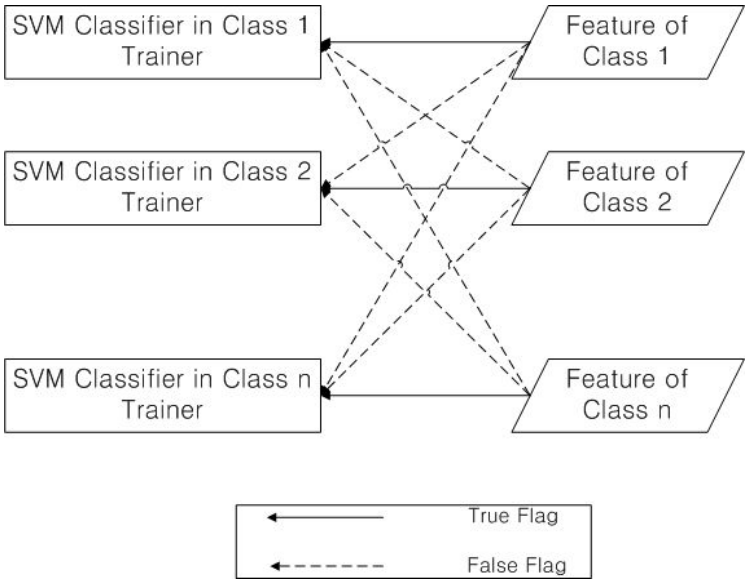


그림 2.9 SVM 분류기의 학습 방법

SVM 분류기는 각 부류별로 한 개씩 생성되고, 각 특징 벡터는 고유 출력값을 가지고 학습기에 입력된다. 이 과정에서 해당 분류기와 같은 번호를 가지면 True 플래그를, 이외의 값은 False 플래그를 가지고 학습된다. 학습이 끝난 분류기에 분류하고자 하는 데이터가 입력되면 입력 데이터는 모든 분류기에 입력되어 유효 데이터와 그렇지 않는 데이터를 판단하게 되는 형식이다. 1:1 방식이라 데이터베이스가 구축된 상태에서 해당되는 데이터를 찾아내는 데엔 좋은 성능을 나타내지만 이외의 경우 오분류율이 높다는 문제가 생겨 본 논문에서는 Adaboost 알고리즘을 사용하였다

2. Adaboost 알고리즘

Adaboost는 적응적 부스팅(Adaptive boosting)을 말한다. 이 방법의 핵심은 분류 규칙을 순차적으로 생성하고, 이전의 분류 규칙을 적용하여 얻은 관측값에서 샘플 데이터의 분포를 재조정한다. 학습 초기에 샘플 데이터의 가중치는 동일한 상태에서 시작하지만, 매 라운드가 진행될수록 이전의 분류 규칙을 적용하여 얻은 관측값으로, 오분류 데이터에는 높은 가중치를 부여하고, 반대로 정분류된 데이터에는 낮은 가중치를 부여하는 방식으로 샘플 데이터의 분포를 재조정한다 [9].

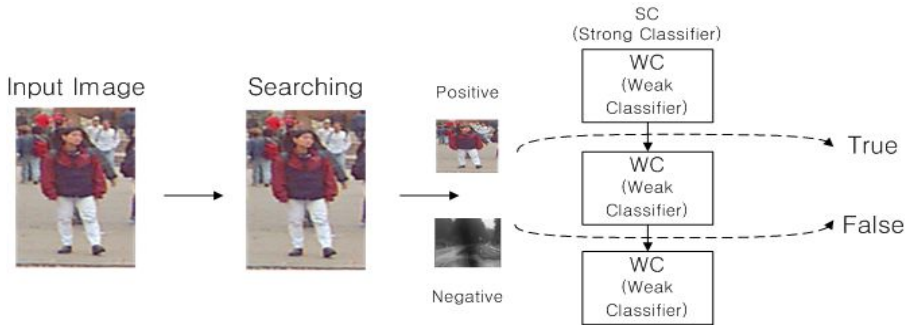


그림 2.10 Adaboost학습 방법

Adaboost 알고리즘은 그림 2.10과 같이 약한 분류기(Weak classifier)를 선형적 결합으로 학습하여 강한 분류기(Strong classifier)를 생성하는 것을 목적으로 한다. 하나의 영상에서 보행자를 검출할 경우 일정 크기의 블록이 영상 전체를 돌아가며 보행자를 찾게 된다. 이는 실시간 검출에 있어서 기존의 방법에 비해 뛰어난 성능을 보이며, 복잡하고 다양한 배경에서 강인하게 보행자를 추출 할 수 있는 장점을 가지고 있다. 검출 시 16x16의 최저 창 크기를 설정하며 검출하는데 이 때 HOG특징은 대략 7,000개 이상 존재한다. 전체 특징을 모두 연산하는 것은 상당한 연산량을 필요로 하며 Adaboost 알고리즘은 각각의 특징에 대한 성능 평가를 하고 전체 특징에서 성능이 좋은 특징을 선택하게 된다.

하나의 약한 분류기는 특징 점의 정보, 이진 분류시의 임계값, 특징의 부등호를 정해주는 Parity가 존재한다.

$$hj(x)\begin{cases}1 & \text{if} \quad p_j f_j(x) < p_j \theta_j \\ 0 & otherwise\end{cases} \tag{9}$$

식 (9)에서 f_j는 특징 점의 정보, θ_j 는 이진 분류 임계값 p_j는 특징의 부등호 방향을 나타낸다. 약 분류기의 정보를 이용하여 AdaBoost 알고리즘을 수행하며 HOG를 이용한 약 분류기 집합을 만들어놓게 된다.

Adaboost 알고리즘의 수행 과정은 다음과 같다.

a. 학습에 필요한 영상 집합$(x_1,y_1)...(x_n,y_n)$을 준비한다. 보행자인 영상, 보행자가 아닌 영상에 대해 $y_i = 0,1$ 이다.

b. $y_i = 0,1$ 인 학습 영상 집합 각각에 대해서 가중치 $w_{1,i} = \frac{1}{2m}, \frac{1}{2l}$ 을 초기화 한다. 여기서 m과 l은 보행자인 영상과 아닌 영상의 개수이다.

c. $t = 1,...,T$ 에서

(1) 가중치를 정규화한다. $w_{t,i} = \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}}$ 여기서 $w_{t,i}$는 t번째 약한 분류기에 입력되는 I번째 훈련영상의 가중치를 의미한다. j는 특징점을 의미한다. 특징점에 대해 약한 분류기를 학습시킨다.

(2) 각각의 특징에 대해서 분류기를 학습한다.

(3) 학습된 약한 분류기에서 에러 $\varepsilon_j = \sum_i w_i |F_j(x_i) - y_i|$ 를 계산한다.

(4) 최저의 에러값 ε_j 를 가지는 분류기 F_j 를 선택한다.

(5) 각 학습 집합에 할당된 가중치 $w_{t+1,i} = w_{t,i}\beta_t^{1-e_i}, (\beta_t = \frac{\varepsilon_t}{1-\varepsilon_t})$ 를 갱신한다.

(6) 만약 영상 x_i가 정확하다면 $e_i = 0$, 그렇지 않으면 $e_i = 1$이다.

d. 최종적으로 강한 분류기는 다음과 같다.

$$h(x) = \begin{cases} 1 & \sum_{t=1}^{T} a_i h_i \geq 0.5 \sum_{t=1}^{T} a_t \\ 0 & otherwise \end{cases}$$ 여기서 $a_t = \log \frac{1}{\beta_t}$ 이다.

Adaboost 알고리즘으로 특징을 선택하고 강한 분류기를 만들면, 그 분류기는 보행

자를 효과적으로 검출할 수 있게 된다. 약한 분류기를 생성하는 과정에서 최소 에러를 가지는 약한 분류기를 선택하며 이 분류기에 가중치를 부과한다. 오분류된 훈련 영상은 가중치 w_i를 증가시키고 정분류된 훈련 영상은 가중치 w_i를 감소시킨다. 후반부에 얻어진 특징들은 구별이 어렵기 때문에 가중치를 통해 쉽게 구별할 수 있다. 이와 같이 Adaboost를 이용해 선택된 특징들과 새로 구성되는 강한 분류기에 대한 평가를 동시에 진행하며 강한 분류기가 구성되는 근거를 마련한다.

C. Dalal이 제안한 보행자 추적 알고리즘

일반적으로 대부분의 그림2.11과 같이 보행자 추적 알고리즘은 영상에서 보행자를 검출할 때 특징 파라미터를 사용하여 보행자 검출이 되면 검출된 부분에서부터 보행자 영역을 설정한다. 여기서 특징 파라미터는 보행자 전신에 대한 특징을 의미한다. 이 특징 파라미터는 사람 얼굴이나 차량에 대한 특징 파라미터를 얻는다면 보행자 검출만이 아니더라도 해당 부분에서의 응용이 가능한 방법이다.

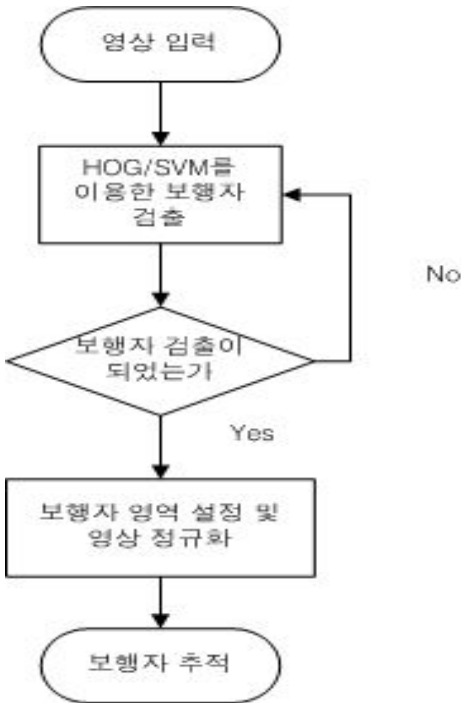


그림 2.11 일반적인 보행자 추적 순서

Dalal이 제안한 방법은 보행자를 검출하는 경우 그림 2.12 와 같은 순서를 따른다. 이 방법에서 보행자의 특징 검출 방법은 HOG를 이용하였으며, 특징 검출 후 추적 방법은 Haar-like 특징을 이용한 방법보다 효율적이라는 실험 결과를 얻었다.

그림 2.12 Dalal이 제안한 HOG/SVM을 이용한 특징학습 및 보행자 분류 과정

Dalal은 그림 2.13과 같이 보행자의 겹침을 개선하고자 칼만필터를 적용하였다. 보행자 전신 영역 중 일부 영역이 유실되는 미약한 겹침현상의 경우 보행자 영역 설정은 실패하지 않는다. 붉은색의 부분에서 주황색 옷을 입은 보행자는 앞에 있는 흰색 옷의 보행자로 인해 팔의 일부분이 유실된 상태이다.

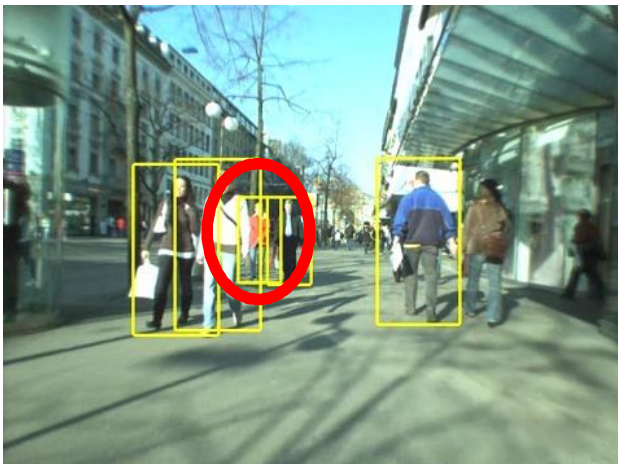


그림 2.13 보행자의 일부분 겹침현상의 개선

그러나 전체 영역이 유실되는 경우엔 해당 보행자의 이동정보를 제공하지 못하고 다시 보행자의 전신이 나타난 경우 앞서 추적했던 보행자인지 아닌지는 구분할 수 없다.

이런 부분을 개선하고자 보행자 영역이 설정된 영역을 기준으로 이전 프레임간의 보행자 위치 정보 즉, 움직임 벡터를 얻어낸다. 이렇게 얻어진 움직임 벡터는 프레임단위로 생성되고 이 값을 누적시켜 보행자가 이동하는 예상 지점을 얻어내 보행자를 검출할 수 있는 방법을 제안한다.

III. 제안하는 보행자 추적 알고리즘

제안하는 방법에서는 그림 3.1과 같은 과정에서 전체 특징을 학습하는 Adaboost를 사용한다. Adaboost는 약한 분류기를 초기에 설정하고 학습하여 최종적으로 강한 분류기를 생성하여 보행자 검출에 사용한다. 검출된 보행자가 겹침현상에 의해서 사라지거나 감지되지 않을 경우 블록 정합 알고리즘을 통해 얻어진 움직임 벡터를 이용한다. 움직임 벡터를 이용 할 경우 보행자 영역과 같은 크기의 윈도우가 보행자가 움직이고 있는 방향으로 예측 이동한다. 보행자가 감지되지 않은 경우에 해당되는 누적 벡터 영역이 이동되며, 겹친 물체와 분리되었을 경우, 기존에 추적하고 있던 보행자를 놓치지 않고 추적 및 감지가 가능하다.

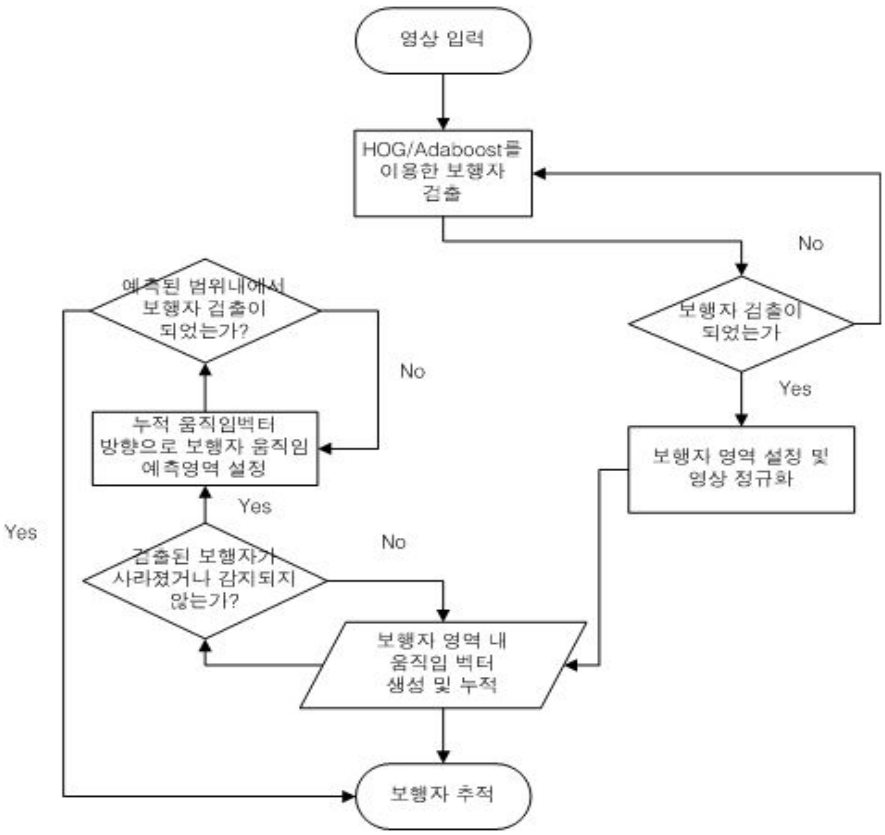


그림 3.1 제안하는 보행자 검출 및 추적 알고리즘

모듈별 설명은 다음과 같다. 영상 입력이 되면 HOG 특징추출을 수행한다. 이렇게 모아진 HOG 특징의 집합을 Adaboost를 이용하여 학습한다. 처음 약한 분류기부터 학습하여 이후에 생성되는 강한 분류기를 통해서 보행자를 검출한다.

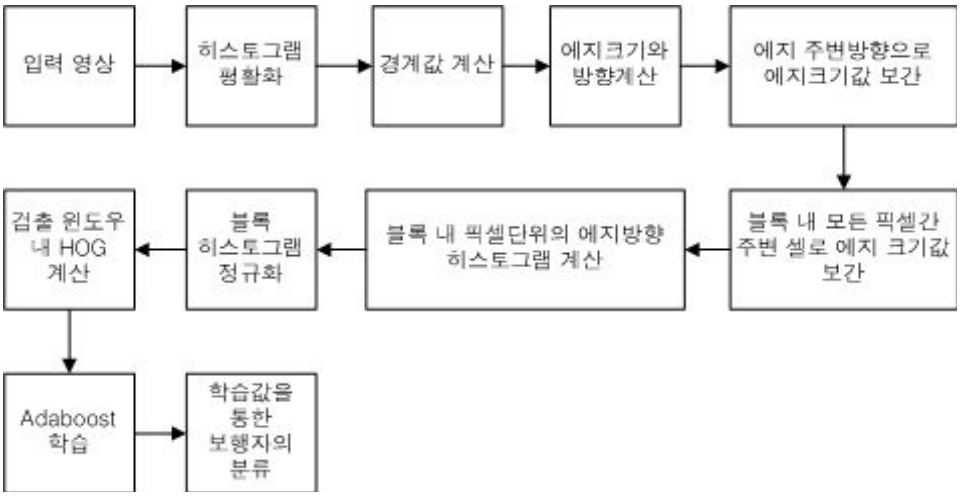


그림 3.2 HOG/Adaboost를 이용한 보행자 특징학습 및 보행자 분류 과정

보행자 영역 설정 및 영상 정규화 부분에서 그림 3.3과 같이 검출된 보행자를 중심으로 녹색의 사각형 범위를 설정한다. 보행자가 초기에 검출되지 않은 경우에는 보행자가 없다고 판단하며, 미리 학습된 데이터를 이용하여 지속적으로 보행자를 추적한다. 만약 보행자가 검출이 되었다면 검출된 보행자를 중심으로 한 보행자 영역을 설정한다.

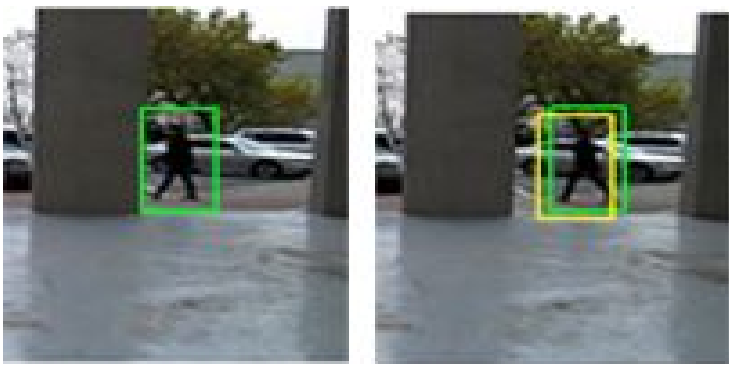


그림 3.3 설정된 보행자 영역의 예

보행자 영역 내 움직임벡터 생성 및 누적 부분에서는 그림3.4와 같이 보행자의 녹색 사각형 범위를 기준으로 블록 정합 알고리즘을 실행한다. 이렇게 생성된 움직임벡터는 프레임단위로 계산되어 누적된다. 이는 보행자의 이동 방향과 평균적 이동 거리를 파악하는데 사용된다. 계산하는 동안 보행자를 지속적으로 추적하게 된다.

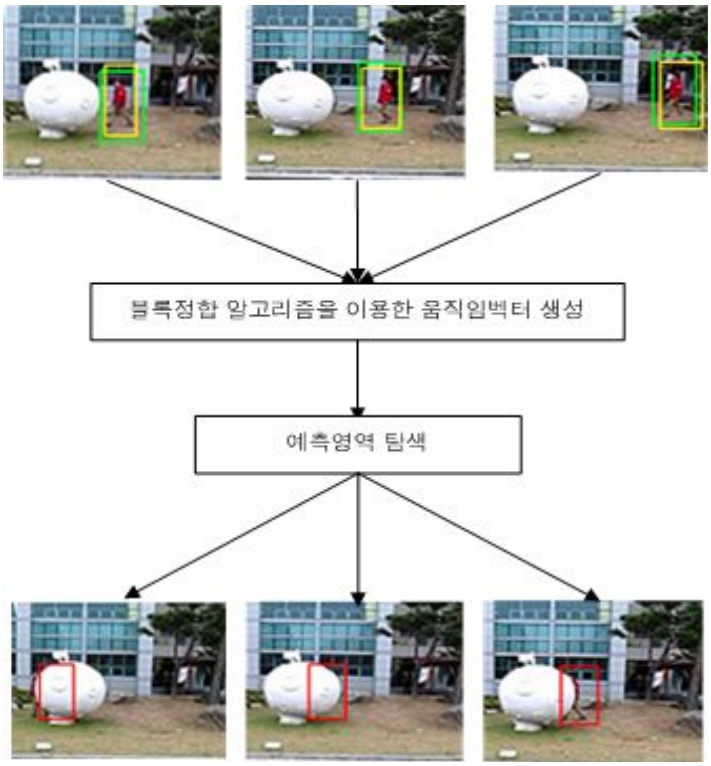


그림 3.4 움직임 벡터 생성 및 예측영역 탐색의 예

만약, 검출된 보행자가 사라졌거나 감지되지 않는 경우에는 누적 움직임벡터 방향으로 보행자의 움직임 예측영역을 설정한다. 보행자가 부분 또는 전체 유실되어 감지되지 않을 경우, 누적 움직임벡터를 이용하여 보행자가 있다고 예측될 수 있는 위치에 붉은색의 사각형 범위를 생성한다. 지속적으로 감지되는 경우에는 계속적으로 움직임 벡터를 누적시킨다.

보행자가 나타나지 않았다면 예측 영역은 계속적으로 이동하게 된다. 예측된 범위에서 보행자 검출이 된 경우 붉은색의 사각형 영역은 사라지고 새로 보행자 영역이 설정되며 추적하게 된다.

IV. 실험 및 고찰

A. 실험 순서

본 논문에서는 제안하는 방법의 성능 평가를 위해 기본적인 HOG 방법과 동일한 셀 크기, 블록 크기 등 동일한 파라미터를 설정한다. 사용된 시스템은 표 4.1과 같다.

표 4.1 실험에 사용된 시스템 사양 및 시뮬레이션 환경

운영체제	Windows 7 enterprise
프로세서	Intel core i7 cpu
메모리	4096MB RAM
그래픽카드	NVIDIA GeForce GTS240
사용환경	Visual studio 2008

또한 제안된 보행자 인식 및 검출 방법을 실험하기 위해 공개된 보행자 데이터인 MIT pedestrian 데이터base, INRIA person 데이터set을 이용하였다. 각각의 보행자 데이터는 MIT 924장, INRIA 1132장이다. 그리고 보행자가 포함되지 않는 비보행자 사진 1000장을 사용하였다. 영상 크기는 제공된 형태의 64x128 크기를 사용한다.

학습 후 얻어진 특징 데이터를 이용해 본 대학원 건물 주변에서 임의로 보행자 동영상을 촬영하여 실험한다. 임의의 동영상은 데이터 A, 데이터 B로 정의한다. 1인에 대한 보행자 검출과 추적을 실행하며 겹침현상이 생기는 상황을 가정하여 촬영되었다.

다음 그림은 학습 데이터로 사용한 보행자 사진 16장과 비보행자 사진 16장을 예로 나타내었다.

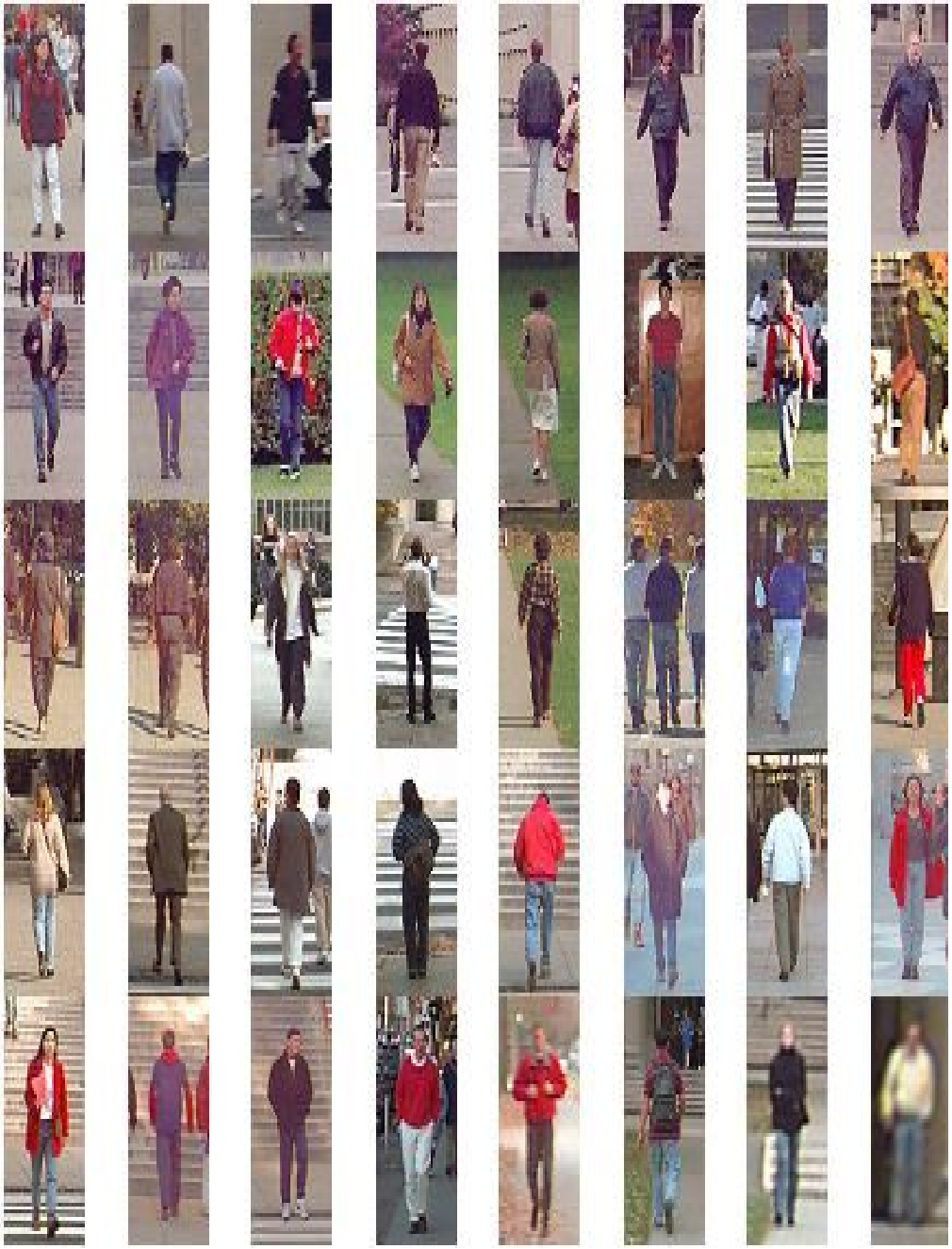


그림 4.1 실험에 사용된 보행자 사진 데이터

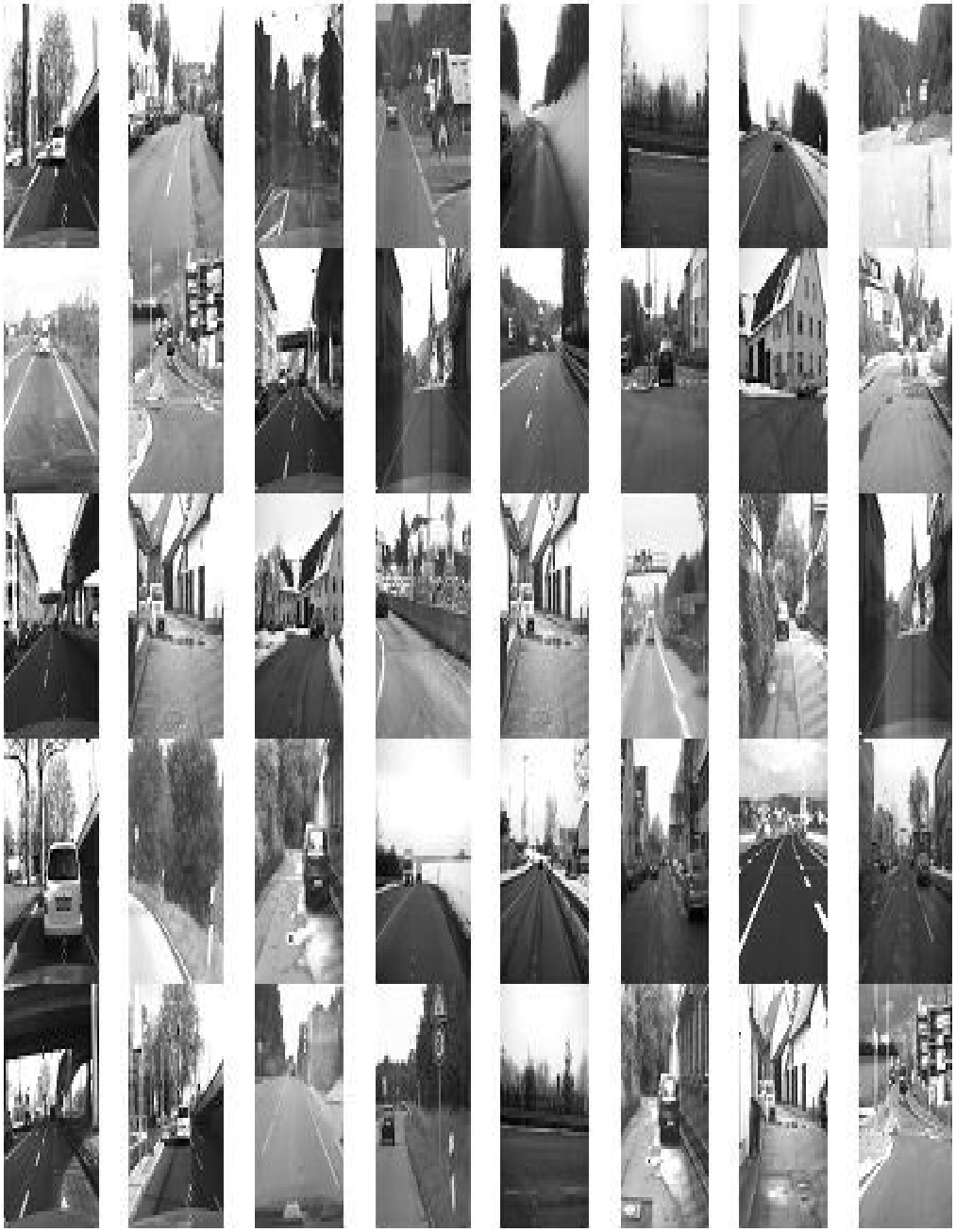


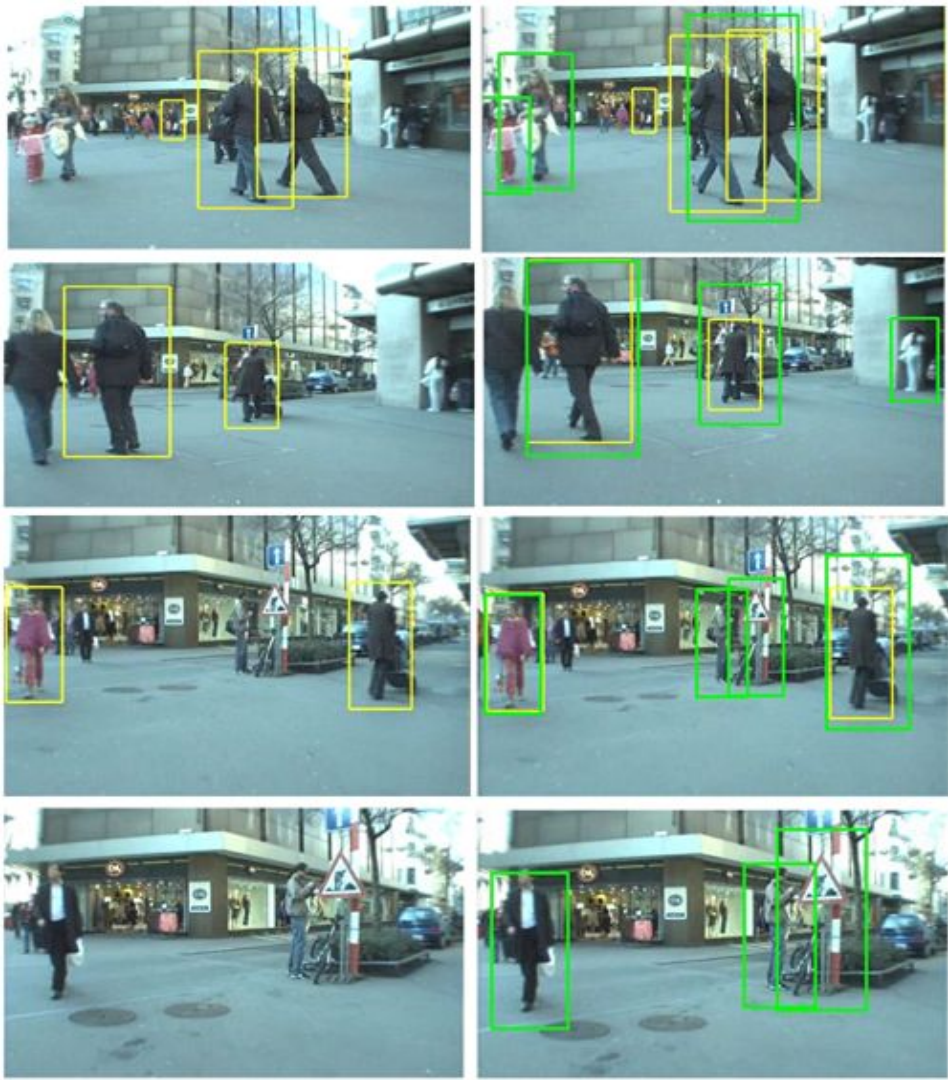
그림 4.2 실험에 사용된 비보행자 사진 데이터

보행자와 비보행자 사진으로부터 HOG 특징벡터를 추출하고, 추출된 데이터들은 Adaboost를 이용해 보행자와 비보행자를 학습 및 분류하였다. 자체적으로 사용한 동영상은 실험실 주변 실외 환경에서 촬영하였으며, 성능평가를 위해 Dalal이 제안한 알고리즘과 성능을 비교하였다.

실험 순서는 다음과 같다. 겹침현상이 일어나는 경우를 가정한 영상을 이용하여 비교가 쉽도록 기존의 방법과 제안하는 방법을 비교 실험한다. 실험 결과는 프레임별로 정리하여 비교하고, 제안하는 방법인 보행자의 움직임 벡터와 누적 움직임벡터를 정리하여 표로 기술한다.

B. 제안하는 방법의 보행자 추적 실험

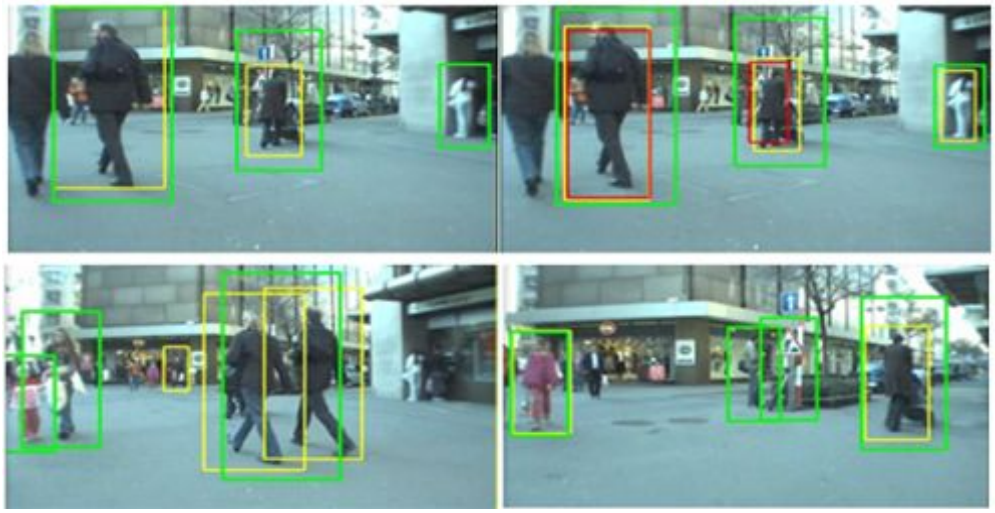
기존의 방법을 이용한 영상 프레임은 그림 4.3의 (a)이며 제안한 방법을 이용한 영상 프레임은 (b)와 같다. 기존의 방법은 노란색 사각형 범위를, 제안한 방법은 녹색 사각형 범위를 지정하여 검출하였다.



(a) (b)

그림 4.3 Dalal의 방법과 제안하는 방법의 보행자 검출 결과 비교

녹색 사각형 범위는 제안한 방법, 노란색 사각형 범위는 기존의 방법을 사용한 보행자 범위이다. 기존의 방법과 비교했을 때 보행자 검출은 그림 4.3와 같이 나아진 것을 확인 할 수 있었다.



(a) (b)

그림 4.4 Dalal의 방법과 제안하는 누적 움직임벡터를 이용한 보행자 검출 결과 비교

그러나 그림 4.4와 같이 보행자가 많은 환경의 영상을 바탕으로 실험한 결과 누적 움직임벡터를 사용한 방법이 기존의 방법보다 나아졌는지 확인하기 어렵다. 그래서 결과 확인이 용이하도록 겹침현상이 일어난다고 가정하는 환경에서 1인 보행자 검출 및 추적을 실험하였다. 실험에 사용된 1인 기준 영상은 그림 4.5, 그림 4.6과 같다.

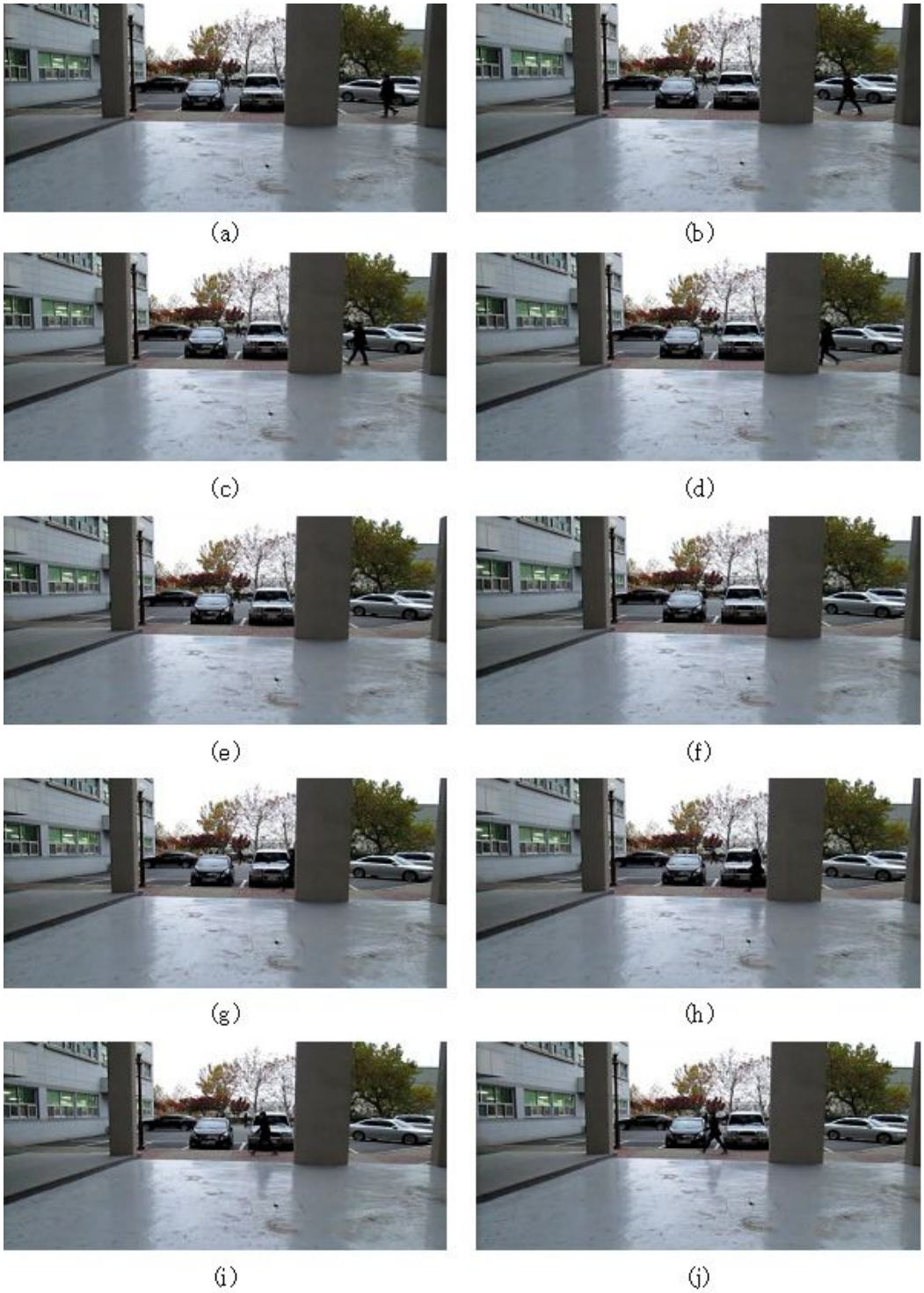


그림 4.5 실험에 사용된 겹침현상을 가정한 프레임단위 영상(데이터 A)

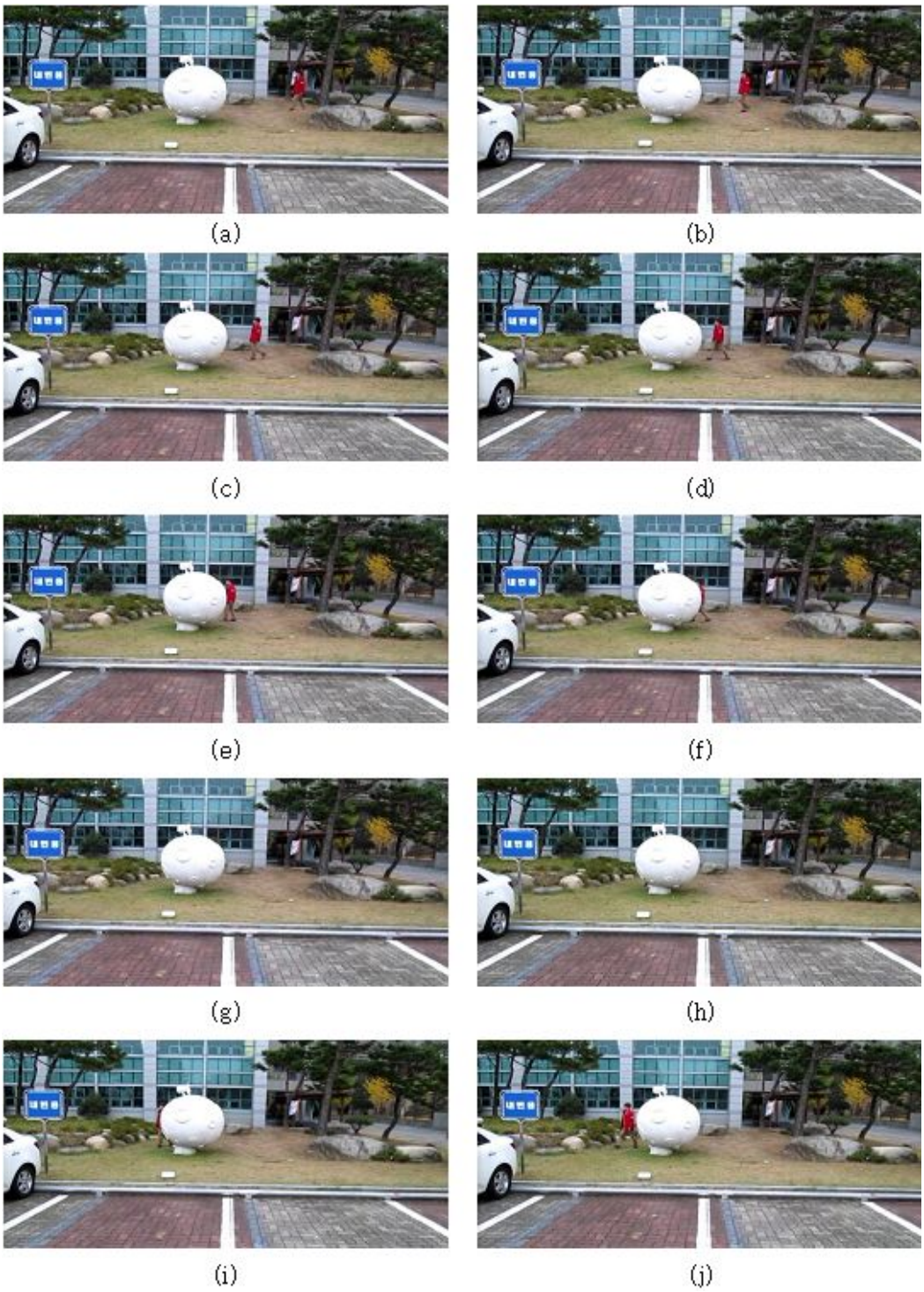


그림 4.6 실험에 사용된 겹침현상을 가정한 프레임단위 영상(데이터 B)

해당 영상은 (a)~(j)의 순서이며 13프레임 단위로 나누어 표시하였다. 13프레임을 기준으로 움직임벡터를 합산한 누적 움직임벡터에 따라 예측 이동 방향을 얻는다. 누적움직임 벡터를 얻기 위해 검출되어 그려진 보행자 영역을 중심으로 이전 프레임과 비교하여 블록 정합 알고리즘을 통해 얻어낸다. 보행자 일부분이 유실되거나 가려지는 경우 예측 이동 방향으로 관심영역을 이동시키며, 보행자가 다시 나타나는 경우 보행자를 재검출한다.

노란색 사각형 영역은 기존의 방법에 대한 보행자 검출을 나타내며 녹색 사각형 영역은 제안한 방법의 보행자 검출을 나타낸다. 누적움직임 벡터를 이용한 예측 이동 영역은 붉은색 사각형 상자로 표시하였다. 제안한 방법을 이용한 보행자 검출 및 추적결과는 아래의 그림 4.7과 4.8에 프레임별로 정리하였다.

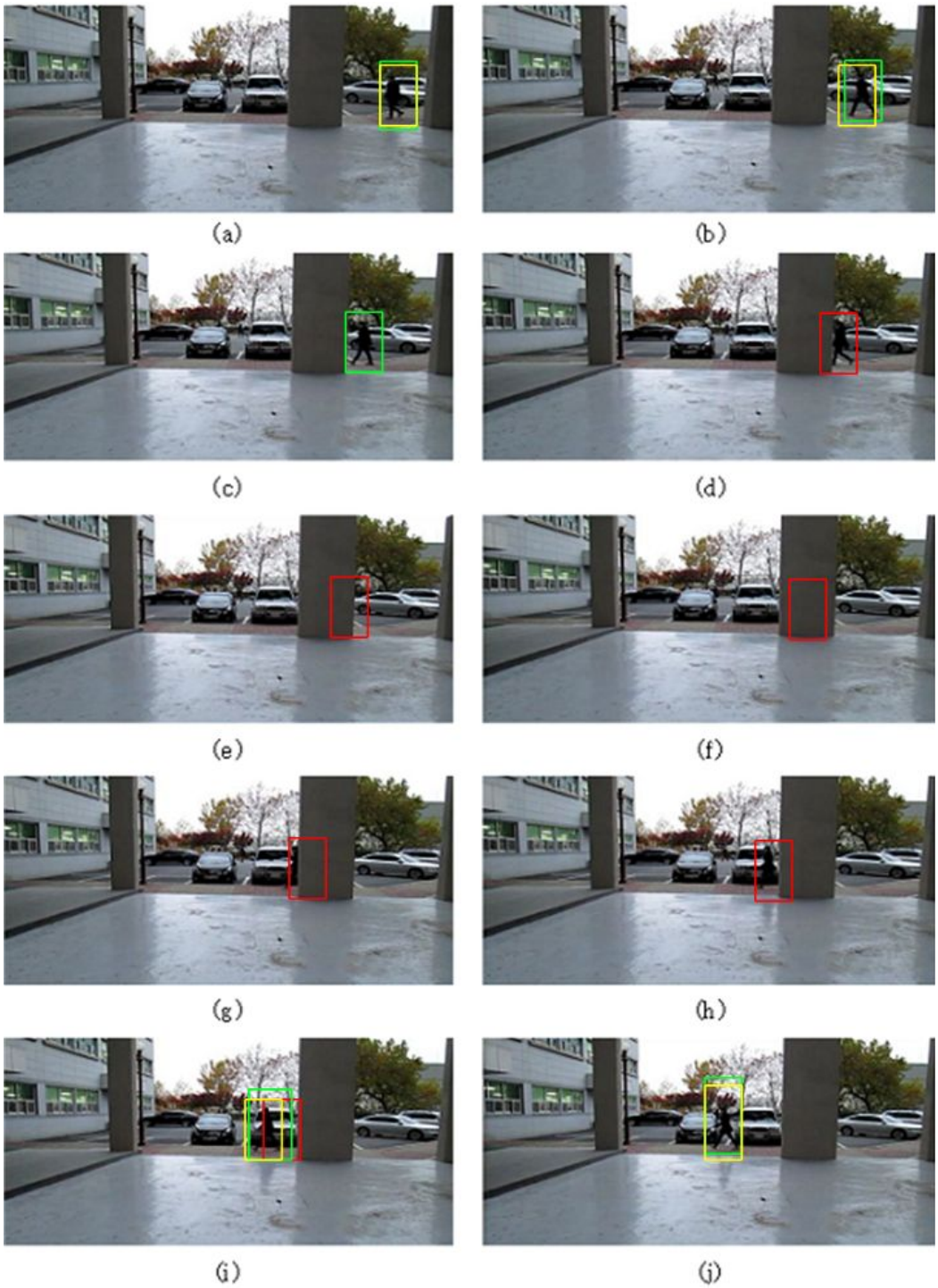


그림 4.7 데이터 A 영상에서의 보행자 검출 및 추적 결과

그림 4.7의 (a)~(c)에서는 보행자의 검출 및 추적이 진행된다. 기존의 방법에서는 (d)프레임부터 보행자 검출이 실패하게 되나, 제안한 방법은 (d)~(i)까지 누적 움직임 벡터로 인한 보행자 방향 예측을 통해 기존에 추적하고 있던 보행자를 놓치지 않는다. (i), (j)에서와 같이 실제 보행자 검출 영역과 예측 영역의 중복된 부분을 확인하면 예측 범위가 실제 보행자 검출 범위와 비슷하다는 것을 확인할 수 있다. 데이터 A에서의 보행자 움직임벡터와 누적 움직임벡터 그리고 이동방향은 아래 표 4.2와 같다.

표 4.2 데이터 A의 (a)~(c)프레임 움직임벡터와 예측 이동 방향

움직임벡터	누적 움직임벡터	예측 이동 방향
(-4, 0)	(-4, 0)	왼쪽
(-6, 0)	(-5, 0)	왼쪽

누적 움직임벡터로 지정한 예측범위는 x, y축을 기준으로 상, 하, 좌, 우 그리고 대각선 방향으로 이동된다. 누적 움직임벡터는 프레임간 블록 정합 알고리즘으로 얻은 움직임벡터를 합산 후 평균을 구해 사용한다. 보행자가 검출된 경우 누적 움직임벡터는 폐기되며 보행자가 검출되는 시점으로부터 다시 계산된다.

보행자의 움직임벡터는 보행자의 이동하는 방향과 현재 이동 속도를, 누적 움직임벡터는 보행자의 평균적인 이동방향과 속도를 나타낸다고 볼 수 있다. 보행자 검출이 실패한 구간을 기준으로 예측 영역은 이동하게 된다. 표 4.2를 보면 데이터 A의 보행자는 평균적으로 x축으로 -5, y축으로 0만큼 이동한다고 할 수 있다. 실제 그림 4.7에서도 왼쪽으로 이동하는 것을 확인할 수 있다.

그림 4.8 데이터 B에서의 보행자 검출 및 추적 결과

그림4.8에서 (a)~(e)에서는 보행자의 검출 및 추적이 진행된다. (e)의 경우 기존의 방법은 보행자 추적에 실패한 것을 확인할 수 있다. (a)~(e)까지 계산된 누적움직임벡터에 따라 보행자 검출에 실패하는 경우에 누적 움직임벡터를 통한 예측 영역이 이동하게 되며 (j)에서 찾아낸 보행자 영역과 예측 영역이 비슷한 위치에 존재하는 것을 확인할 수 있다. 데이터 B에서의 보행자 움직임벡터와 누적 움직임벡터 그리고 이동방향은 아래 표 4.3과 같다.

표 4.3 데이터 B중 (a)~(d)프레임의 움직임벡터와 예측 이동 방향

움직임벡터	누적 움직임벡터	예측 이동 방향
(-8, 0)	(-8, 0)	왼쪽
(-11, 0)	(-9.5, 0)	왼쪽
(-7, -1)	(-8.6, -0.3)	왼쪽
(-4, 0)	(-7.5, -0.25)	왼쪽

실제로 누적 움직임벡터를 사용할 때 소수점 이하의 데이터는 반올림하여 사용한다. 소수점 이하의 값을 버리고 사용해도 값이 미미하므로 결과에 미치는 영향은 전무하다. 표 4.3의 누적 움직임벡터를 통해 데이터 B의 보행자는 평균적으로 x축으로 -7.5, y축으로 -0.25만큼 이동한다. 그림 4.8에서도 보행자는 왼쪽으로 이동하고 있는 것을 확인할 수 있다.

전체적으로 기존의 방법과 비교했을 때의 검출률은 다음 표 4.4와 같다. 성능평가는 CLEAR metrics의 MOTP(Multiple Object Tracking Precision), MOTA(Multiple Object Tracking Accuracy)로 구성되어 있다 [20]. MOTP는 실제 객체의 이동방향과 제안하는 방법의 예측 범위의 오차를 측정하며 MOTA는 객체를 놓친 경우, 오검출의 경우, 검출 불일치의 경우를 합산하여 정확도를 분석하였다.

표 4.4 Dalal의 방법과 제안하는 방법의 보행자 검출 비교표(단위 : %)

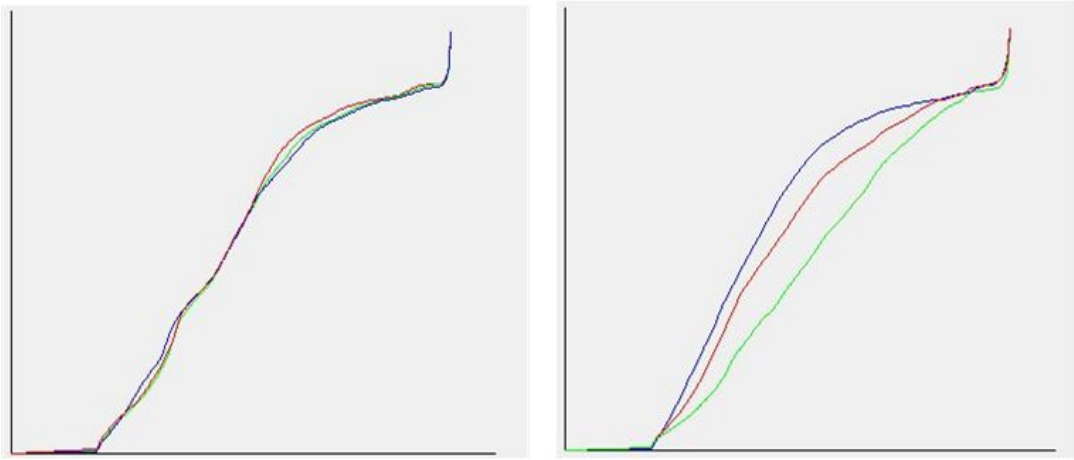
Video 데이터 / Method	eth		데이터 A		데이터 B	
	MOTP	MOTA	MOTP	MOTA	MOTP	MOTA
Dalal et al.	73.3	78.1	72.2	78.9	76.5	81.3
Proposed Method	78.7	80.1	83.4	85.1	84.6	87.2

데이터 A와 데이터 B에서 예측영역 내의 보행자는 비슷한 위치에 존재한다. 그리고 이전의 보행자가 검출되는 프레임에서의 보행자의 의상의 색상과 움직임벡터를 이용한 예측 범위가 의미하는 이동속도를 바탕으로 이전에 추적하던 보행자라는 것을 판단 할 수 있다.

(a) (b)

그림 4.9 원본영상과 검출영상의 보행자 의상 색상 및 위치 비교

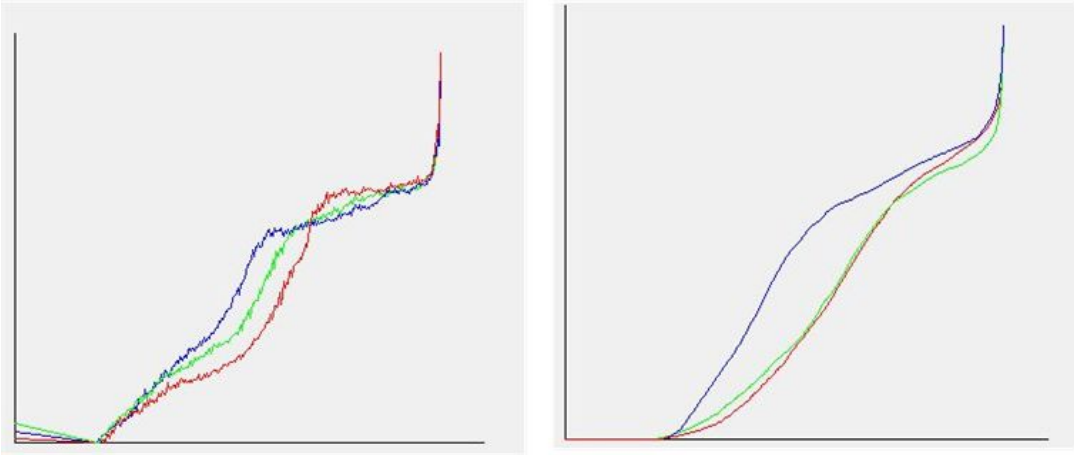
그림 4.9에서 원본영상에서의 보행자의 의상 색과 같은 프레임에서의 보행자 검출 결과를 보면 같은 보행자가 검출되었다는 것을 확인 할 수 있다. 그리고 이후 예측영역에서 나타난 보행자 또한 원본영상의 보행자와 검출 결과의 보행자와 같다는 것을 확인 할 수 있다. 아래 그림 4.10과 4.11은 검출된 영역의 RGB히스토그램을 나타낸다.



(a) (b)

그림 4.10 데이터 A의 기존 보행자(a)와 예측 영역의 보행자(b)의 RGB색상 분포 곡선

위의 그림에서 이전의 보행자가 검출되었을 때의 RGB분포와 예측영역에서의 보행자 RGB분포를 나타낸 것이다. 그림 4.10(a)의 기존 보행자의 RGB 색상 분포와 4.10(b)의 예측 영역에서 나타난 보행자의 RGB색상 분포 곡선이 비슷하다는 것을 확인 할 수 있다. 마찬가지로 그림 4.11(a)의 기존 보행자의 RGB 색상 분포와 4.11(b)의 예측 영역에서 나타난 보행자의 RGB색상분포곡선 또한 비슷하다는 것을 확인 할 수 있다. 이것은 보행자의 의상의 색이 예측영역에서 검출된 보행자의 의상 색과 비슷하다고 판단할 수 있는 근거가 된다.



(a) (b)

그림 4.11 데이터 B의 기존 보행자(a)와 예측 영역의 보행자(b)의 RGB 색상 분포 곡선

색상 분포 곡선은 보행자 설정 영역의 배경에도 영향을 받는다. 보행자 주변의 배경 색상으로 인하여 두 보행자 영역 간 RGB색상곡선이 완전히 같을 수는 없다. 그러나 보행자 중심으로 설정되는 영역으로부터 얻어진 보행자의 색상 분포 곡선이 비슷하다는 것은 기존의 보행자인지 아닌지 판단 할 수 있는 좋은 근거가 된다.

V. 결론 및 향후 연구과제

본 논문에서는 동영상에서 움직이는 보행자를 HOG특징 추출과 Adaboost학습 과정을 통한 파라미터를 생성하여 검출하고 추적하였다. 겹침현상에서 발생하는 보행자 검출과 추적에 실패하는 경우를 개선하고자 보행자 영역 주변에서 움직임벡터를 계산하였다. 그리고 프레임별로 그 값을 누적시켜 얻어진 누적 움직임벡터를 이용해 겹침현상에서 보행자 움직임 방향을 예측하는 시스템을 제안하였다.

실험을 통하여 겹침현상에서도 기존의 방법보다 보행자 움직임 예측과 추적이 잘 이루어짐을 증명하였으며, 기존의 유사한 방법의 실험결과와 비교를 통해 제안한 방법이 기존의 방법보다 일부 성능이 개선됨을 확인하였다.

추적하고 있던 보행자 전신 영역 중 일부 영역이 사라지는 경우 검출이 불가능하여 추적에 실패하는 상황에서 누적 움직임벡터를 병합하여 이용함으로써 해당 문제를 보완할 수 있다 생각된다.

그러나 다수의 보행자가 존재하며 보행자끼리 서로 겹치는 환경에서 제안한 방법이 어느정도 움직임 예측은 가능하였다. 그러나 다수의 보행자 상황에서 예측 범위에 나타난 보행자가 기존에 추적하고 있는 보행자인지 새로운 보행자인지 구별하는 것은 거의 불가능했다. 이런 환경에서도 강건하게 보행자 이동 방향을 예측하고 검출 할 수 있는 방법이 필요하다.

이 방법은 디스크립터를 이용한 보행자 객체의 개별 ID를 이용하여 구별이 가능할 것으로 판단된다. 향후 이런 방법을 구체적으로 연구하며 보행자 범위를 구별하는 방법과 제안하는 방법을 좀 더 개선시켜 다수의 보행자가 겹치는 상황에서도 각각의 보행자를 구별할 수 있는 연구를 진행하고자 한다.

참고 문헌

[1] M. Valera and S. A. Velastin, "Intelligent Distributed Surveillance Systems : a review", Proceeding of Vision'05 Image and Signal Processing, Vol. 152, pp. 92-204, 2005.

[2] Markus Enzweiler and M. Gavrila, "Monocular Pedestrian Detection : Survey and Experiments", IEEE Trans on Pattern Analysis and Machine Intelligence, Vol. 31, No. 12, Dec. 2009.

[3] T. Sakai, M. Nagao, and S. Fujibayashi, "Line Extraction and Pattern Recognition in a Photograph", Pattern Recognition, Vol. 1, pp. 233-248, 1969.

[4] Lim, J. S. and Kim, K. H, "Multiple Pedestrians Detection and Tracking using Color Information form Moving Camera", Journal of KIPS : Korea Information and Applications, Vol. 11, pp. 317-326, 2004.

[5] A. Broggi, M. Bertozzi, and A. Fascioli, "Shape based Pedestrian Detection", Proceeding of the IEEE Intelligent Vehicle Symposium 2000, pp. 215-220, 2000.

[6] M. Mä0hlisch, M. Oberländer, O. Löhlein, D. Gavrila, and W. Ritter, "A Multiple Detector approach to low-resolution for Pedestrian Recognition", Proceeding of the IEEE Intelligent Vehicles Symposium 2005, pp. 23-28, 2005.

[7] Osuna, E. Freund, and R.Girosit, "Support vector Machines : Training and applications", AI Memo 1602, Massachusetts Institude of Technology, Mar. 1997.

[8] P. Viola, M. Jones, and D. Snow, "Detecting Pedestrians Using Patterns of Motion and Appearance", International journal of Computer Vision, Vol. 63, No. 2, pp. 153-161, 2005.

[9] S. Paisitkriangkrai, C. Shen, and J. Zhang, "Fast Pedestrian Detection Using a Cascade of Boosted Covariance Features", IEEE Trans Circuit and System for Video Technology, Vol. 18, No. 8, pp. 1140-1151, 2008.

[10] C, Papageorgiou and T. Poggio, "A Trainable System for Object Detection," International Journal of Computer Vision, Vol. 38, No. 1, pp. 15-33, 2000.

[11] C. Curio, J. Edelbrunner, T. Kalinke, C. Tzomakas, and W. Vom Seelen, "Walking Pedestrian Recognition", IEEE Trans on Intelligent Transportation System, Vol 1, No. 3, pp. 155-163, 2000.

[12] N. Dalal and B. Triggs, "Histogram of Oriented Gradients for Human Detection", INRIA Rhone-Alps, Montbonnot 38334, France, 2005.

[13] Q. Zhu, S. Avidan, M. Chen Yeh, and K. TingCheng, "Fast Human Detection using a Cascade of Histograms of Oriented Gradients", Mitsubishi Electronic Research Laboratories, 2006.

[14] A. Shashua, Y. Gdalyahu, and G. Hayun, "Pedestrian Detection for Driving Assistance Systems : Single-frame Classification and System Level Performance", Mobileye, Italy, 2004.

[15] P. Dollar, C. Wojek, B. Schiele, and P. Perona, "Pedestrian Detection: An evaluation of the state of the art", IEEE Trans on PAMI, 2012.

[16] P. F. felenszwalb, R. B. Girshick, D. Mcallester, and D. Ramanan, "Object Detection with Discriminatively Trained Part Based Models", IEEE Trans on PAMI, 2010.

[17] R. Benenson, M. Mathias, R. Timofte, and L. V. Gool, "Pedestrian Detection at 100 frames per second", In Proceeding of CVPR, 2012.

[18] L. M. Po and W. C. Ma, "A Novel four-step Search Algorithm for Fast Block Motion Estimation", IEEE Trans on Circuits and Systems for Video Technology, Vol. 6, No. 3, pp. 313-317, 1996.

[19] C. Zhu, X. Lin, and L. P. Chau, "Hexagon-based Search Pattern for Fast Block Motion Estimation", IEEE Trans on Circuits and Systems for Video Technology, Vol. 12, No. 5, pp. 349-355, 2002.

[20] K. Bernardin and R. Stiefelhagen, "Evaluating Multiple Object Tracking Performance : The CLEAR MOT metrics", IN EURASIP JIVP, 2008.