이동 카메라 영상에서 컬러 정보를 이용한 다수 보행자 검출 및 추적

임 종 석† · 김 욱 현††

요 약

본 논문에서는 이동 카메라에서 취득한 영상에서 컬러 정보를 이용하여 다수의 보행자를 검출하고 특정 보행자를 추적하는 방법을 제안한다. 먼저 연속한 동영상 입력에 대해 BMA(Block Matching Algorithm)을 이용하여 움직임 벡터를 추출하고 움직임 보상을 한 후 차 영상을 생성한다. 다음은 이진 영상으로 변환한 후 불필요한 잡음 등을 제거하고, 프로젝션을 수행하여 보행자를 검출한다. 만약 검출된 보행자가 서로 인접하거나 겹쳐졌을 경우 RGB 컬러 정보를 이용하여 분리시킨다. 검출된 다수의 보행자로부터 특정 보행자를 추적하기 위해 보행자 가운데 영역의 RGB 컬러 정보를 이용하여 추적한다. 제안된 방법에 대하여 비디오 카메라로 녹화한 영상을 컴퓨터에서 입력받아 검출과 추적 실험을 수행한 결과, 검출 성공률이 97%, 검출 실패율이 3%로 나타났고 추적 또한 우수함을 입증하였다.

Multiple Pedestrians Detection and Tracking using Color Information from a Moving Camera

Jong-Seok Lim† · Wook-Hyun Kim††

ABSTRACT

This paper presents a new method for the detection of multiple pedestrians and tracking of a specific pedestrian using color information from a moving camera. We first extract motion vector on the input image using BMA. Next, a difference image is calculated on the basis of the motion vector. The difference image is converted to a binary image. The binary image has an unnecessary noise. So, it is removed by means of the proposed noise deletion method. Then, we detect pedestrians through the projection algorithm. But, if pedestrians are very adjacent to each other, we separate them using RGB color information. And we track a specific pedestrian using RGB color information in center region of it. The experimental results on our test sequences demonstrated the high efficiency of our approach as it had shown detection success ratio of 97% and detection failure ratio of 3% and excellent tracking.

키워드 : 컬러 정보(Color Information), 움직임 벡터(Motion Vector), 움직임 보상(Motion Compensation), 블록 매칭 알고리즘(BMA), 차 영상(Difference Image)

1. 서 론

최근 동영상에서 움직이는 물체에 대한 정보를 추출하기 위한 연구가 활발히 진행되고 있다. 움직이는 물체의 추출은 군사, 교통, 의학, 생물학, 공학, 교육 등 기타 여러 분야에서 응용될 수 있다. 특히, 보행자 검출 및 추적은 시내를 주행하는 인공지능 자동차, 무인 자동차 시스템[1]을 만들기 위한 필수적인 요소라 할 수 있다. 즉, 위험한 교통 상황을 미리 감지하여 서로 충돌을 피할 수 있다. 뿐만 아니라 보행자 검출 및 추적은 보안 또는 감시 시스템, 횡단보도 신호제어 시스템[2-4] 등 실세계의 다양한 응용 분야에서 사용될 수 있다.

보행자 검출은 보행자를 추적하기 위한 전 단계로 보행자가 제대로 검출됨으로써 보행자 추적이 원활하게 이루어질 수 있다. 본 논문에서는 하나의 보행자가 아닌 다수의 보행자를 검출하고 검출된 보행자로부터 특정 보행자를 추적하는 방법을 제안한다.

일반적으로 움직이는 물체를 검출하기 위한 영상 획득 방법은 두 가지가 있다. 첫째는, 하나의 카메라로부터 얻은 단일 영상에서 연속된 영상을 이용하는 방법이다. 둘째는, 두 개의 카메라로부터 얻은 스테레오 영상에서 두 영상의 깊이 차를 이용하는 방법이다[5].

단일 영상을 이용하는 방법에는 상관 기반 방법[6], 광류 기반 방법[7], 체형 기반 방법[8], 움직임 기반 방법[9-11], 차

† 정 회 원 : 영남대학교 대학원 컴퓨터공학과
†† 정 회 원 : 영남대학교 전자정보공학부 교수
논문접수 : 2003년 7월 24일, 심사완료 : 2004년 4월 27일

영상 분석 방법[3, 12] 등이 있다. 상관 기반 방법과 광류 기반 방법은 카메라 움직임이 있어도 동작하지만 물체의 외형이 변하는 유동적인 물체를 검출하기 어렵다. 체형 기반 방법은 보행자를 인식하기 위해 체형 특징에 의존한다. 그래서 단일 영상으로부터 움직이거나 정지한 보행자를 모두 검출할 수 있다. 그러나 이 방법은 보행자의 외형인 자세, 유동적인 움직임, 밝기, 옷, 가려짐 등 광범위한 변화 상태를 고려해야하는 단점이 있다. 움직임 기반 방법은 인간에게 고유한 규칙적인 특징이나 동작 패턴을 사용한다. 하지만 단점은 보행자의 발이나 무릎은 규칙적인 특징을 추출하기 위해 분명해야 한다. 또한 제약이 없고 주변을 배회하고, 회전하고, 점프하는 등과 같은 복잡한 동작을 수행하는 보행자와 정지한 보행자는 검출할 수 없는 단점이 있다. 차영상 분석 방법은 주위 환경 변화에 강건하고 계산이 간단하며 다양한 유형의 보행자를 검출하는 장점이 있으나 정지한 보행자나 카메라 이동이 있을 경우 보행자를 검출할 수 없는 단점이 있다.

본 논문에서는 하나의 카메라로부터 획득한 컬러 영상으로부터 블록매칭(BMA)을 수행하여 움직임 벡터를 구한다. 블록매칭은 전역적 탐색을 수행하면 시간이 많이 걸리므로 특정 영역에 대해서만 탐색을 수행하여 움직임 벡터를 구한다. 이 움직임 벡터 값을 이용하여 움직임 보상을 수행한 후 RGB 컬러 값의 차를 이용하여 차 영상을 구한다. 그리고 이 차 영상을 이진 영상으로 변환한 후 불필요한 잡음 등을 제거한다. 이제 보행자를 검출하기 위해 프로젝션을 수행한다. 그러나 프로젝션 방법은 보행자가 서로 떨어져 있는 경우에는 유용하게 보행자를 검출하지만 보행자가 서로 인접해 있거나 겹쳐진 경우에는 검출할 수 없는 단점이 있다. 따라서 본 논문에서는 RGB 컬러 정보를 이용하여 이를 해결할 수 있는 방법을 제안한다. 본 논문에서 제안한 방법은 카메라 이동이 있는 경우에도 보행자를 검출할 수 있고 서로 인접해 있거나 약간 겹쳐진 경우에도 보행자를 검출할 수 있는 장점이 있다.

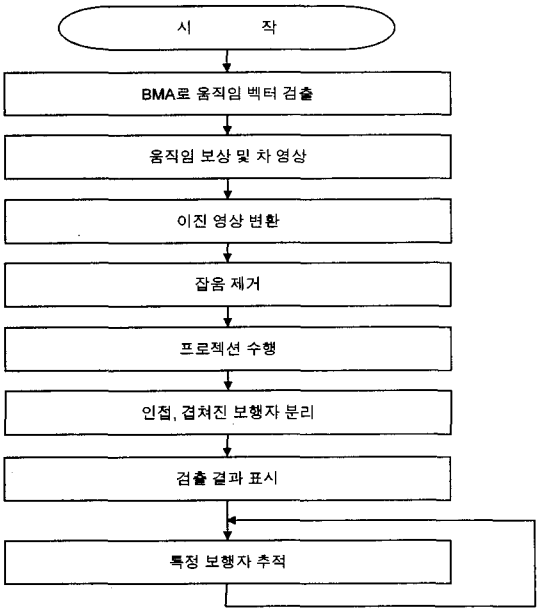
또한 본 논문에서는 검출된 다수의 보행자로부터 특정 보행자를 추적하는 방법을 제안한다. 기존의 이동물체 추적 방법은 부분 외곽선 정보[13]를 이용하거나 거리변환을 이용하여 특정 물체를 탐색하는 방법[14, 15], Hausdorff 정합을 이용하는 방법[16, 17] 등이 있다. Hausdorff 정합법은 검출된 이동물체의 윤곽선 영상을 모델로 하여 Hausdorff 거리를 모델과 탐색 영상 사이의 동일성 판별의 척도로 사용하여 추적함으로써 추적대상의 형태변화에 적응할 수 있다.

본 논문에서는 보행자 좌우 가운데 영역의 RGB 컬러 정보를 이용하여 보행자를 추적하는 방법을 제안한다.

2. 보행자 검출 및 추적 시스템 개요

본 논문에서 제안하고 있는 보행자 검출 및 추적 시스템은 (그림 1)과 같다.

먼저 비디오 영상에서 두 개의 프레임을 입력받아 블록매칭 알고리즘(BMA)을 수행한다. 이 알고리즘은 카메라 이동이 발생한 경우 카메라 이동 량을 검출하는데 사용한다. 그러므로 영상에서 전체를 탐색하지 않고 특정 영역에 대해서만 수행한 후 움직임 벡터를 구한다. 그리고 이 움직임 벡터를 이용하여 현재 프레임에 움직임 보상을 수행한다. 이렇게 움직임 보상된 프레임과 이전 프레임을 컬러 레벨 차를 이용하여 차 영상을 구한다. 그러면 움직이는 물체가 검출된다. 물체가 검출된 영상을 이진 영상으로 변환하고 변환된 이진 영상에는 프로젝션을 수행하는데 불필요한 점 등의 잡음이 존재하므로 이를 제거하고 프로젝션을 수행한다. 그리고 이 프로젝션을 인간의 체형 정보를 바탕으로 분석하여 보행자를 검출한다. 그러나 보행자가 서로 인접하거나 겹쳐진 경우에는 프로젝션 방법으로는 검출할 수 없으므로 현재 프레임의 RGB 컬러 정보를 이용하여 분리시킨다. 검출된 보행자로부터 특정 보행자를 추적하기 위하여 보행자 좌우 가운데 영역의 RGB 컬러 정보를 이용하여 추적한다.



(그림 1) 보행자 검출 및 추적 시스템 구성도

3. 제안한 보행자 검출 및 추적 알고리즘

3.1 BMA를 이용한 움직임 벡터 검출

카메라 이동이 있을 경우 배경이 변하므로 이 상태에서

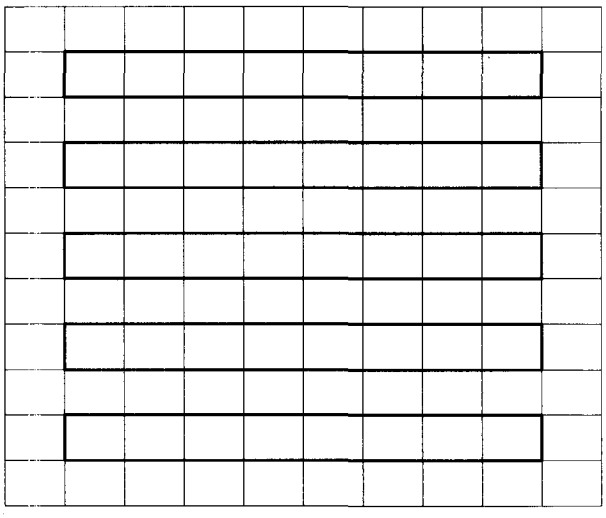
두 프레임간 차 영상을 구하면 움직이는 물체뿐만 아니라 다른 배경 영역도 검출된다. 그래서 두 프레임 간의 움직임 벡터를 검출하고 이 벡터를 이용하여 움직임 보상을 한 후 차 영상을 구하면 영상 내의 움직이는 물체만을 검출할 수 있다.

움직임 벡터는 블록 매칭 알고리즘(BMA)을 이용하여 구한다. 즉 이전 프레임 F_{n-1}의 구획을 상하좌우로 움직이고 현재 프레임 F_n의 차분 총합 D가 최소로 되는 방향(dx, dy)를 움직임 벡터로 구한다. 움직임 벡터를 구하는 식은 다음과 같다.

$$D=\sum_i\sum_j |F_n(x_i,y_j)-F_{n-1}(x_i+dx,y_j+dy)| \quad (1)$$

식 (1)에서 i, j는 영상의 크기를 나타내는 첨자이다.

움직임 벡터를 구할 때 구획의 크기는 16×16으로 설정하고 구획을 움직이는 최대량은 x방향, y방향 모두 8로 설정한다. 영상 내의 모든 구획에 대하여 움직임 벡터를 구하면 시간이 많이 걸리므로 특정 구획에 대해서만 움직임 벡터를 구한다. 즉 영상을 (그림 2)와 같이 5개의 라인으로 골고루 설정하고, 이 5개 라인 각각에 대해서 맨 왼쪽 첫 번째 구획과 맨 오른쪽 마지막 구획을 제외한 구획에 대해서만 움직임 벡터를 구한다.



(그림 2) 움직임 벡터 검출 영역

움직임 벡터가 각 구획에 대해서 구해지면 움직임 보상을 수행하기 위해 대표 벡터를 구한다. 대표 벡터는 x 벡터 값은 각 구획의 x 벡터 값 중에서 가장 큰 값으로 설정하고, y 벡터 값은 각 구획의 y 벡터 값 중에서 가장 큰 값으로 설정한다.

(그림 3)은 식 (1)을 적용하여 구한 움직임 벡터를 나타내고 있다.

(그림 3) 검출한 움직임 벡터(x = 5, y = 1)

3.2 움직임 보상(Motion Compensation) 및 차 영상 (Difference Image)

움직임 벡터가 구해지면 대표 벡터를 이용하여 현재 프레임에 움직임 보상을 수행한다. 대표 벡터 값이 x = 5, y = 1이라고 하면 움직임 보상(Motion Compensation)은 현재 프레임에서 x방향으로 -5만큼, y방향으로 -1만큼 영상을 이동시키면 된다. 그러나 영상 자체를 이동시키면 시간이 많이 걸리므로 차 영상을 구할 때 움직임 벡터를 적용하는 것이 시간을 더 절약할 수 있어 좋다.

차 영상(Difference Image) 방법은 연속적인 두 프레임의 그레이 레벨 차를 이용하여 영상 내에 움직이는 물체를 검출하는 전통적인 방법이다. 이 방법은 배경이 변하지 않는 영상일 경우 잘 동작하지만 카메라 이동에 의한 배경 화면이 변할 때 움직이는 물체를 검출하지 못하는 단점이 있다. 본 논문에서는 움직임 벡터를 이용하여 움직임 보상을 수행하므로 카메라 이동에 의한 배경 화면이 변해도 움직이는 물체를 검출할 수 있다. 본 논문에서는 RGB 컬러 값을 이용하여 차 영상을 구한다.

일반적으로 차 영상을 구하는 식은 다음과 같다.

$$d(i,j) = |F_n(i,j)-F_{n-1}(i,j)| \quad (2)$$

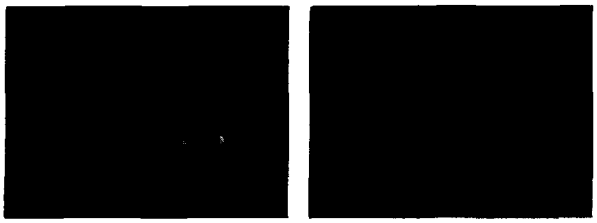
식 (2)에서 d(i, j)는 RGB값이 모두 동일한 영상으로 움직임이 있는 영역을 나타낸다.

한편, 움직임 벡터를 이용하여 움직임 보상을 한 후 차 영상을 구해야 하므로 식 (2)를 다음과 같이 수정한다.

$$d(i,j) = |F_n(i+x,j+y)-F_{n-1}(i,j)| \quad (3)$$

식 (3)에서 x, y는 움직임 벡터 값이다.

(그림 4)는 움직임 보상을 수행한 후의 차 영상과 움직임 보상 없이 수행한 차 영상을 나타낸다. 움직임 보상을 수행한 후 차 영상을 구한 (그림 4)(a)가 그렇지 않은 (그림 4)(b) 보다 더 배경이 단순함을 알 수 있다.



(a) 움직임 보상 수행 후 차 영상 (b) 움직임 보상 없이 수행한 차 영상

(그림 4) 움직임 보상 및 보상 없이 수행한 차 영상 비교

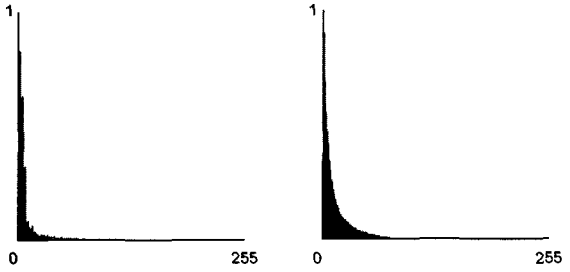
3.3 프로젝션을 이용한 보행자 검출

차 영상을 구했으면 프로젝션을 이용하여 보행자를 검출한다. 프로젝션을 수행하려면 이진 영상으로 변환해야 한다.

이진 영상(Binary Image)은 프로젝션 수행 속도를 증가시키는데 있다. 영상을 이진 영상으로 변환하는 식은 다음과 같다.

$$B(x,y)=\begin{cases} I(x,y)\geq Th \text{ 이면} & 255 \\ \text{그렇지 않으면} & 0 \end{cases} \quad (4)$$

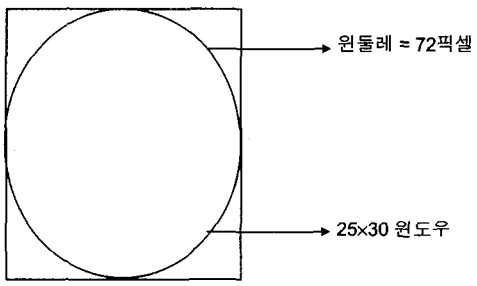
식 (4)에서 B(x, y)는 이진 영상을 의미하고, I(x, y)에는 움직임 보상을 수행한 차 영상이 대입된다. Th는 임계값으로써 차 영상의 RGB 평균값을 이용하여 히스토그램을 작성한 결과를 근거로 사용하였다. (그림 5)는 여러 종류의 차 영상을 이용한 RGB 평균값 히스토그램을 나타낸 것이다. 가로축은 RGB 평균값이고 세로축은 가장 큰 빈도수로 나눈 정규화된 값이다. 그리고 최대값(Max)은 빈도수가 0이 아닌 가장 큰 값이다. 히스토그램에서 알 수 있듯이 최대값의 중간값을 이진 영상의 임계값으로 사용하면 가장 좋은 결과를 얻을 수 있으므로 본 논문에서는 40을 사용하였다.



(a) Max = 75 (b) Max = 82

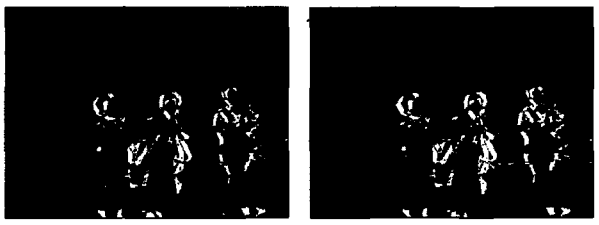
(그림 5) RGB 평균값 히스토그램

변환된 이진 영상에는 불필요한 잡음 등이 많이 존재하므로 이를 제거해야 프로젝션을 수행하는데 어려움이 없고 쉽게 보행자를 검출할 수 있다. 잡음 제거는 고립점 제거, 직선 제거, 영역 제거 등이 있는데 본 논문에서는 영역 제거를 통해서 불필요한 잡음을 제거한다. 영역 제거는 n×n 크기의 슬라이딩 윈도우를 이동하면서 윈도우 영역 내에 있는 픽셀 개수를 카운트하여 임계값 보다 적으면 잡음으로 간주하여 그 영역 내의 모든 픽셀들을 제거하는 방법이다. 본 논문에서는 슬라이딩 윈도우의 크기를 가로 25, 세로 30 크기로 설정하였다. 이것은 실험에서 사용하는 영상 내의 보행자 얼굴에 해당하는 크기이다. 그리고 임계값은 (그림 6)과 같이 윈도우 내에 내접하는 원을 그렸을 때 원둘레의 크기에 해당하는 픽셀 개수로 설정하였다. 영역 제거를 수행할 때 윈도우의 크기는 보행자의 크기에 따라 적절하게 변경해야한다.



(그림 6) 슬라이딩 윈도우와 임계값

(그림 7)은 차 영상을 이진 영상으로 변환한 영상과 영역 제거를 수행한 영상을 나타낸다.



(a) 이진영상(Th = 40) (b) 영역 제거 영상

(그림 7) 이진 영상과 영역 제거 영상

잡음 제거가 완료되면 프로젝션을 수행하여 프로젝션 히스토그램을 생성한다. 프로젝션은 수평 방향과 수직 방향으로 수행한다. 이진 영상에서 각각의 방향으로 0 보다 큰 픽셀 값(픽셀 값 255)의 개수를 카운트하는 방식으로 프로젝션 히스토그램을 만든다. 다음은 각각 수평 방향과 수직 방향의 프로젝션 히스토그램을 만드는 식이다.

$$H_i=\sum_{j=0}^{n-1} B(i,j) \quad (5)$$

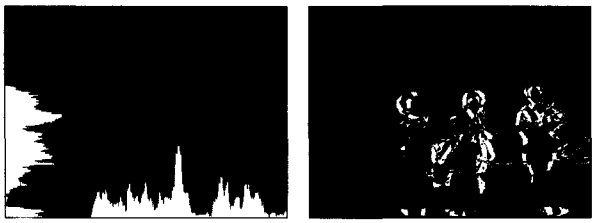
$$V_j=\sum_{i=0}^{m-1} B(i,j) \quad (6)$$

식 (5)와 식 (6)에서 B(i, j)는 이진 영상, i는 0에서 m-1, j는 0에서 n - 1까지이고, m은 영상의 세로 크기, n은 영상

의 가로 크기이다.

이진 영상에 프로젝션을 수행하여 프로젝션 히스토그램을 생성하고 나면 이 프로젝션 히스토그램을 인간의 체형 정보를 바탕으로 분석하여 보행자를 검출한다. 그러나 이 방식은 보행자가 서로 인접해 있거나 겹쳐져 있는 경우, 또는 카메라 이동이 심하여 배경이 검출된 경우 보행자를 검출하지 못하는 단점이 있다.

(그림 8)은 프로젝션 히스토그램 영상과 이 영상을 이용하여 보행자를 검출한 결과를 나타낸다. (그림 8)(a)는 수평 방향과 수직 방향 프로젝션 히스토그램을 나타낸다. 가로축은 수직 방향 프로젝션이고 세로축은 수평 방향 프로젝션이다. (그림 8)(b)는 카메라 이동이 심하여 배경에 있는 자동차 때문에 보행자를 제대로 검출하지 못함을 보여주고 있다.



(a) 프로젝션 히스토그램 (b) 보행자 검출 영상

(그림 8) 프로젝션 히스토그램 영상과 보행자 검출 영상

따라서 본 논문에서는 프로젝션 방법으로 보행자 검출이 불가능할 경우 현재 프레임의 RGB 컬러 정보를 이용하여 보행자를 검출한다.

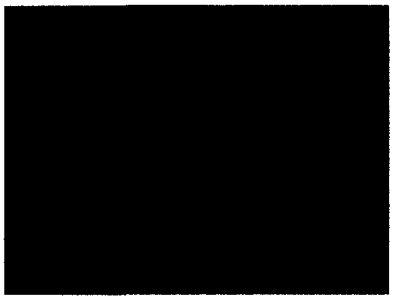
3.4 RGB 컬러 정보를 이용한 보행자 검출

프로젝션 방법으로 보행자 검출이 불가능할 경우 RGB 컬러 정보를 이용하여 보행자를 검출한다. 프로젝션을 수행하고 나면 보행자 영역을 검출할 수 있다. 검출된 보행자 영역에서 보행자의 머리부분이 있는 맨 왼쪽과 맨 오른쪽 위치를 구한다. 그리고 맨 왼쪽의 세로위치와 맨 오른쪽의 세로위치를 구한 후 이 위치를 포함하는 영역에 대해 RGB 컬러 정보를 이용하여 히스토그램을 작성한다. 만약 세로 길이가 20픽셀이 안되면 20픽셀로 설정한다. 다음은 현재 프레임에서 앞에서 구한 영역에 대해 RGB 컬러 값에 대한 영역 히스토그램을 구하는 방법이다.

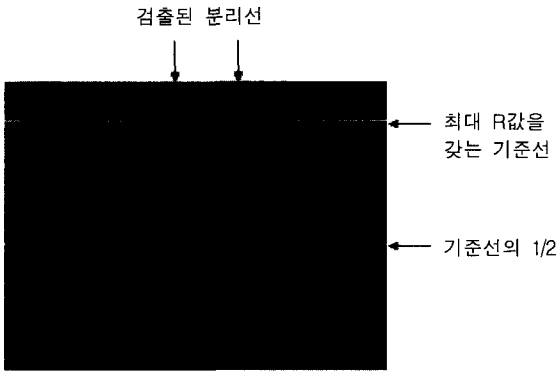
① 구하고자 하는 영역의 첫 번째 라인 상에 있는 각 픽셀의 R값, G값, B값을 표시한다.
② 두 번째 라인부터 마지막 라인까지 ①번 과정을 반복한다.

여기서, RGB 값은 0에서 255사이의 값을 가지므로 이를 히스토그램으로 간단하게 표시하기 위해 최대크기가 127이 되도록 조정한다.

(그림 9)(a)는 (그림 8)에서 검출한 보행자 영상의 보행자 영역을 이용하여 구한 영역 히스토그램을 나타낸다. (그림 9)(a)를 살펴보면 R성분과 B성분으로 보행자를 분리시킬 수 있음을 알 수 있다.



(a) 영역 히스토그램



(b) 검출한 보행자 분리선

(그림 9) 영역 히스토그램과 보행자 좌우 위치

다음은 영역 히스토그램을 이용하여 보행자를 분리시키는 방법이다.

① 영역 히스토그램에서 가장 큰 값을 갖는 R성분의 위치를 찾는다.
② ①에서 찾은 위치를 기준으로 영역 히스토그램의 1/2 되는 위치를 구한다.
③ ②에서 찾은 위치에서부터 가로 방향으로 진행하면서 R성분, G성분, B성분이 모두 0 인 부분을 카운트한다.
④ ③에서 구한 값이 4이상이면 구한 값의 1/2되는 위치를 분리점으로 설정한다.
⑤ 만약 검출한 분리점의 세로선 상에 B성분이 존재하면 이 분리점은 제외한다.
⑥ 분리점이 너무 많거나 하나도 검출하지 못하면 B성분을 이용하여 다시 분리점을 찾는다.

(그림 9)(b)는 위 방법으로 검출한 보행자의 분리점을 세로 선으로 나타낸 것이다. 분리선이 세 개로 나타났으나 이는 최종 과정에서 제거할 수 있다.

보행자를 분리하는 분리선이 검출되었으므로 최종적으로 보행자 주위에 바운딩 박스(Bounding Box)를 만들기 위해 (그림 8)(b)의 보행자 검출 영상과 (그림 9)(b)의 보행자 분리선를 결합한다. 그리고 그 상태에서 각각의 보행자 상하좌우 공백을 제거하여 위치를 다시 조정한다.

(그림 10)은 최종적으로 보행자를 검출한 영상을 나타낸다. 카메라 이동에 의한 잡음 때문에 세 번째 보행자가 정확하게 검출되지 못하였다.

(그림 10) 보행자 검출 영상

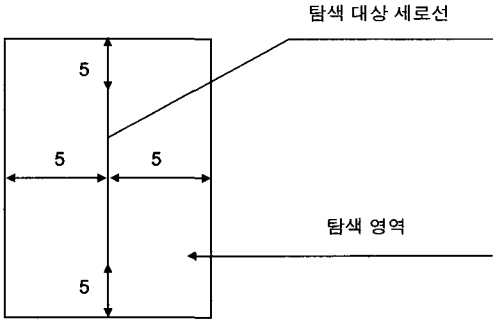
3.5 RGB 컬러 정보를 이용한 보행자 추적

보행자 검출이 끝나면 검출된 보행자로부터 특정 보행자를 RGB 컬러 정보를 이용하여 추적한다. 보행자 추적에 사용하는 RGB 컬러 정보는 (그림 11)과 같다. 각 보행자마다 존재하는 세로 선이 있는 영역의 RGB 컬러 값을 이용하여 보행자를 추적한다. 이 세로 선 영역은 바운딩 박스를 만든 직후 결정하게된다. 즉, 가로 방향으로 처음 만나는 픽셀에서부터 마지막 픽셀까지의 중간 지점을 찾고, 그 지점에서부터 세로 방향으로 바운딩 박스의 아래쪽 지점까지이다.

(그림 11) 추적 RGB 영역

추적 대상의 보행자가 결정되면 그 보행자의 세로 선 영역에 있는 RGB 컬러 값을 다음 프레임에서 최대로 매칭되는 곳을 찾는다. 이때 탐색 영역은 세로 선의 상하좌우 5픽셀 범위를 대상으로 한다. 현재 프레임에서 탐색 대상인 세로 선 영역의 RGB 컬러 값과 다음 프레임의 탐색 영역 내에서 세로 선 영역의 RGB 컬러 값의 차를 구한다. 이 차의 합이 최소가 되는 세로 선 영역을 기준으로 새로운 바운딩 박스를 생성한다. 이 바운딩 박스를 새로운 탐색 영역으로 설정한 후 위의 과정을 반복하므로써 계속 추적하게된다.

(그림 12)는 추적 시 사용하는 탐색 대상과 탐색 영역을 나타낸다.



(그림 12) 탐색 대상과 탐색 영역

4. 실험 및 고찰

본 논문에서 사용된 실험 영상은 실세계에 존재하는 도시 영상을 이용하였다. 비디오 카메라를 이용하여 손으로 들고 촬영한 동영상을 320×240 컬러 영상으로 프레임 단위로 캡쳐하였다. 이 영상을 Pentium Ⅳ 1.4Ghz 프로세서의 윈도즈2000 프로페셔널 버전 운영체제에서 Visual C++ 6.0을 이용하여 구현하였다.

실험은 보행자 검출과 보행자 추적의 2단계로 진행하였다. 먼저 보행자 검출 알고리즘의 성능은 검출 성공률, 검출 오류률, 검출 실패율을 가지고 평가하였다. 실험에 사용한 영상의 프레임 수는 총200개로 보통 보행자, 느린 보행자, 빠른 보행자, 뛰는 보행자와 같은 4종류의 영상을 가지고 실험하였다.

식 (7), 식 (8), 식 (9)는 각각 검출 성공률, 검출 오류률, 검출 실패율을 나타낸다.

$$\sum_{\text{전체프레임}} \frac{\text{검출한 보행자수}}{\text{프레임 당 보행자수}} \times 100\% \quad (7)$$

$$\sum_{\text{전체프레임}} \frac{\text{검출 못한 보행자수}}{\text{프레임 당 보행자수}} \times 100\% \quad (8)$$

$$\sum_{\text{전체프레임}} \frac{\text{잘못 검출한 보행자수}}{\text{프레임 당 보행자수}} \times 100\% \quad (9)$$

실험에서 보행자 검출 성공은 보행자로 판단되는 보행자

를 검출했을 경우 검출 성공으로 판단한다. 즉 보행자가 서로 겹쳐진 경우 두 보행자의 머리를 구분할 수 있는 경우에만 두명의 보행자로 판단하고 그렇지 않은 경우는 한 명의 보행자로 판단한다.

제안한 방법으로 실험한 결과 검출 성공률은 97%이고, 검출 오류율은 0%이며, 검출 실패율은 3%로 아주 우수한 성능을 보였다. 검출에 실패한 3개의 프레임은 두 명의 보행자가 너무 가까이 있어 한 명의 보행자로 검출했기 때문이다.

<표 1>, <표 2>, <표 3>, <표 4>는 각각 보통 보행자, 느린 보행자, 빠른 보행자, 뛰는 보행자 영상에 대한 보행자 검출 결과를 나타내고 있다. 표에서 FN은 프레임 번호, PN은 프레임에 존재하는 보행자수, SPN은 검출에 성공한 보행자수, NPN은 검출하지 못한 보행자수, EPN은 잘못 검출한 보행자수를 의미한다.

〈표 1〉 보통 보행자 검출 실험 결과

FN	PN	SPN	NPN	EPN
1	4	4	0	0
5	4	4	0	0
10	4	4	0	0
15	3	3	0	0
20	3	3	0	0
25	2	2	0	0
30	2	1	1	0
35	4	4	0	0
40	4	4	0	0
45	4	4	0	0
50	4	4	0	0

〈표 2〉 느린 보행자 검출 실험 결과

FN	PN	SPN	NPN	EPN
51	3	3	0	0
55	3	3	0	0
60	3	3	0	0
65	3	3	0	0
70	3	3	0	0
75	2	2	0	0
80	2	2	0	0
85	4	4	0	0
90	4	3	1	0
95	4	4	0	0
100	4	4	0	0

〈표 3〉 빠른 보행자 검출 실험 결과

FN	PN	SPN	NPN	EPN
101	2	2	0	0
105	2	2	0	0
110	2	2	0	0
115	3	3	0	0
120	3	3	0	0
125	3	3	0	0
130	3	3	0	0
135	4	4	0	0
140	4	4	0	0
145	4	4	0	0
150	4	4	0	0

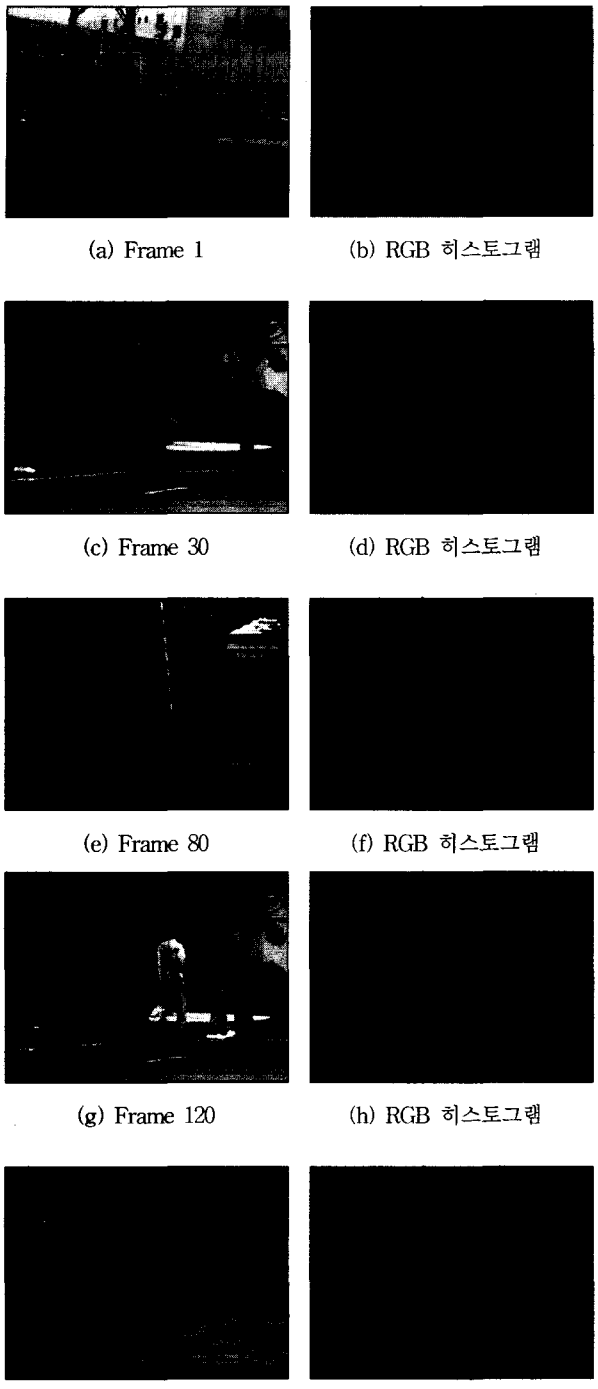
〈표 4〉 뛰는 보행자 검출 실험 결과

FN	PN	SPN	NPN	EPN
151	3	3	0	0
155	3	3	0	0
160	3	3	0	0
165	3	3	0	0
170	3	3	0	0
175	3	3	0	0
180	4	4	0	0
185	4	4	0	0
190	4	4	0	0
195	4	4	0	0
200	4	4	0	0

(그림 13)은 제안한 방법으로 보행자를 검출한 결과 영상을 나타낸 것이다. 왼쪽 열은 보행자 검출 영상을 나타내고, 오른쪽 열은 프로젝션 방법으로 검출하지 못한 영역에 대한 RGB 히스토그램을 나타낸다. 그리고 (그림 13)(a), (그림 13)(c)는 보통 보행자, (그림 13)(e)는 느린 보행자, (그림 13)(g)는 빠른 보행자, (그림 13)(i)는 뛰는 보행자 검출 실험 결과이다. 여기서 (그림 13)(c)는 두 명의 보행자를 한 명으로 검출한 예로 이는 두 보행자가 너무 가까이 있어 검출에 실패하였다.

두 번째로 보행자 추적은 보행자 검출에 의해 검출된 영상에서 좌측 보행자부터 하나씩 순서대로 추적 실험을 하였다. 제안한 보행자 추적 알고리즘은 우수한 성능을 보여주었다.

추적 정확도는 완전탐색기법에 의하여 추적한 위치를 최



(a) Frame 1 (b) RGB 히스토그램

(c) Frame 30 (d) RGB 히스토그램

(e) Frame 80 (f) RGB 히스토그램

(g) Frame 120 (h) RGB 히스토그램

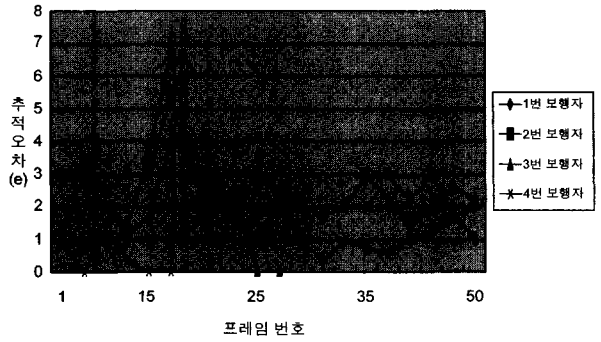
(i) Frame 80 (j) RGB 히스토그램

(그림 13) 보행자 검출 결과 및 RGB 히스토그램

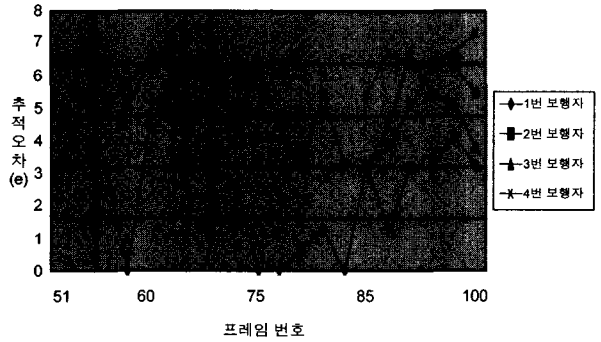
적의 위치로 가정하고, 제안한 방법으로 추적하였을 때 이 두 위치의 오차를 구하여 평가하였다. 완전탐색을 하여 구한 보행자 바운딩 박스의 좌측 위 모서리의 좌표를 (x, y)라 하고, 제안한 방법에 의하여 구한 대응하는 좌표를 (x′, y′)라고 하면 그 오차 e는 다음 식 (10)과 같이 정의한다.

$$e = \sqrt{(x-x')^2+(y-y')^2} \qquad (10)$$

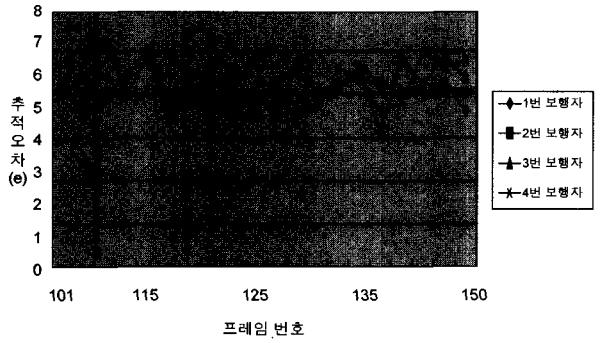
(그림 14)에서 (그림 17)까지는 각각 보통 보행자, 느린 보행자, 빠른 보행자, 뛰는 보행자 검출 영상에서 특정 보행자를 추적한 결과 및 오차를 그래프로 나타낸 것이다.



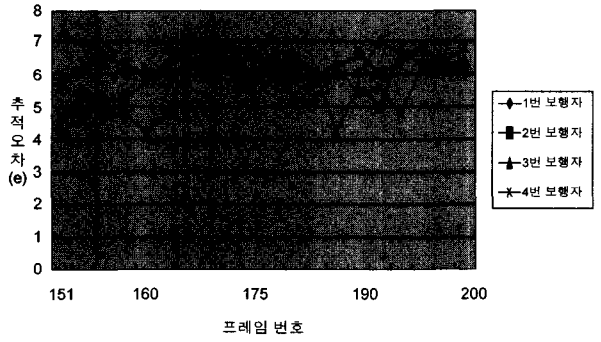
(그림 14) 보통 보행자 추적 오차 그래프(평균오차 2.01)



(그림 15) 느린 보행자 추적 오차 그래프(평균오차 1.43)



(그림 16) 빠른 보행자 추적 오차 그래프(평균오차 2.07)



(그림 17) 뛰는 보행자 추적 오차 그래프(평균오차 2.97)

그래프에서 볼 수 있는 바와 같이 보통 보행자와 느린 보행자 보다 빠른 보행자와 뛰는 보행자에서 오차가 더 크게 나타났음을 알 수 있다. 그러나 제안한 추적 알고리즘은 완전탐색방법으로 탐색한 좌표와 아주 근사한 값을 가지며, 전체 평균오차는 2.12로 매우 높은 추적 정확도를 보임을 알 수 있다.

5. 결론 및 향후 과제

본 논문에서는 이동 카메라에서 취득한 영상에서 다수의 보행자를 검출하고 검출된 보행자 영상에서 특정 보행자를 추적하기 위한 컬러정보기반 알고리즘을 제안하였다. 제안한 알고리즘은 초기에 프로젝션 방법을 이용하여 보행자 영역을 추출하였으며, 그 영역에 대해 RGB 컬러정보를 이용한 히스토그램을 생성하여 보행자를 검출하였다. 그리고 보행자들이 서로 겹치거나 카메라 이동에 의한 잡음 등으로 배경이 검출되어도 효과적으로 보행자를 검출하였다. 또한 검출된 보행자 영상에서 특정 보행자의 가운데 영역의 RGB 컬러정보를 이용하여 추적하였다.

실제 영상에 대한 실험 결과, 제안한 알고리즘은 보통 보행자, 느린 보행자, 빠른 보행자, 뛰는 보행자와 같은 4종류의 영상 200개의 프레임에서 검출 성공률 97%, 검출 오류률 0%, 검출 실패율 3%의 우수한 성능을 보였다. 그리고 검출된 보행자 영상에서 특정 보행자를 추적하여 전체 평균오차 2.12의 매우 높은 추적 정확도를 보였다.

참 고 문 헌

[1] U. Franke, D. Gavrila and S. Goerzig, "Autonomous Driving Approaches Downtown to Appear," IEEE Expert (special issue on Vision-based Driving Assistance in Vehicles of the Future), 1997.

[2] C. Wohler, J. K. Aulaf and T. Portner, U. Franke, "A Time Delay Neural Network Algorithm for Real-time Pedestrian Detection," Proc. of the IEEE Intelligent Vehicles Symposium '98, (stuttgart, Germany), pp.247-251, Oct., 1998.

[3] M. yachida, M. Asada and S. Tsuji, "Automatic Analysis of Moving Image," IEEE Trans. Pattern Anal. Mach. Intell, Vol.PAMI-3, No.1, pp.12-20, 1981.

[4] J. Agbinya and D. Rees, "Multi-object Tracking in Video," Real-Time Imaging 5, pp.295-304, 1999.

[5] L. Zhao and C. Thorpe, "Stereo- and Neural Network-Based Pedestrian Detection," Proc. ITSC '99, pp.5-10, 1999.

[6] H. Inoue, T. Tachikawa and M. Inaba, "Robot Vision System with a Correlation Chip for Real-time Tracking, Optical flow and Depth Map Generation," Proc. of the IEEE International Conference on Robotics and Automation, pp.1621-1626, 1992.

[7] S. Yamamoto, Y. Mae and Y. Shirai, "Real-time Multiple Object Tracking based on Optical Flows," Proc. of the Robotics and Automation, Vol.3, pp.2328-2333, 1995.

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

[9] H. Mori, N. M. Charkari and T. Matsushita, "On-Line Vehicle and Pedestrian Detection based on Sign Pattern," IEEE Trans. on Industrial Electronics, Vol.41, No.4, pp.384-391, Aug., 1994.

[10] S. A. Niyogi and E. H. Adelson, "Analyzing and Recognizing Walking Figures in xyt," Proc. of the IEEE Conference on Computer Vision and Pattern Recognition, pp.469-474, 1994.

[11] S. A. Niyogi and E. H. Adelson, "Analyzing Gait with Spatiotemporal Surfaces," IEEE Workshop on Motion of Non-Rigid and Articulated Objects, pp.64-69, Austin, 1994.

[12] J. S. Lim and W. H. Kim, "Multiple Pedestrians Tracking Using Difference Image and Projection Histogram," Proc. of the International Conf. on Imaging Science, Systems and Technology(CISST '02), Vol.1, pp.329-334, 2002.

[13] 조영석, 이주신, "부분 외곽선 정보를 이용한 이동물체 추적 알고리즘", 정보처리학회논문지B, 제8-B권 제5호, pp.539-548.

[14] D. M. Gavrila and V. Philomin, "Real-time Object Detection Using Distance Transforms," Proc. of IEEE ICIV, Stuttgart, Germany, pp.274-279, 1998.

[15] D. M. Gavrila, "Pedestrian Detection from a Moving Vehicle," Proc. of European Conference on Computer Vision, Dublin, Ireland, pp.37-49, 2000.

[16] D. Huttenlocher, D. Klanderman and A. Rucklidge, "Comparing Images using the Hausdorff Distance," IEEE Trans. on PAMI, Vol.15, No.9, pp.850-863, Sep., 1993.

[17] D. Huttenlocher, R. H. Lilien and C. F. Olson, "View-based Recognition using an Eigenspace Approximation to the Hausdorff Measure," IEEE Trans. on PAMI, Vol.21, No.9, Sep., 1999.

임 종 석

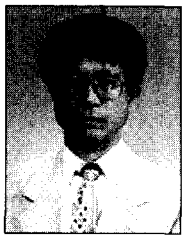
e-mail : robertlim@yumail.ac.kr

1991년 계명대학교 물리학과(학사)

1996년 대구가톨릭대학교 전산통계학과 (이학석사)

2004년 영남대학교 대학원 컴퓨터공학과 (공학박사)

관심분야 : 동영상처리, 패턴인식, 컴퓨터비전



김 욱 현

e-mail : whkim@yumail.ac.kr

1981년 경북대학교 전자공학과(학사)

1983년 경북대학교 대학원 컴퓨터공학과 (공학석사)

1993년 일본 쓰꾸바 대학 공학연구과 (공학박사)

1983~1993년 한국전자통신 연구원 선임연구원

1994년~현재 영남대학교 전자정보공학부 교수

관심분야 : 시각정보처리, 패턴인식, 화상처리