이동물체 검출을 위한 행렬필터 알고리즘

최승욱°, 허화라*, 이장명
부산대학교 전자공학과, 송호대학 정보산업계열*

A Moving Object Detecting Algorithm Using a Matrix Filter

SungYug Choi, HwaRa Hur, JangMyung Lee
Dept. of Electronics Engineering, Pusan Univ. , Dept. of Information Industry, Songho College*

요 약

현재의 영상정보를 이용한 이동물체 검출 알고리즘에서는 물체를 인식하는데 많은 처리시간을 소비한다. 이는 물체의 특징을 사용하여 대상 물체를 일치시키기 위해 대량의 컨볼루션 처리를 하기 때문이다. 따라서, 본 논문에서는 움직이는 물체에 대한 효율적인 궤적 추적 알고리즘의 하나로 행렬필터를 제시하고, 이를 적용한 어플리케이션을 통하여 이를 검증하려 한다.

1. 서론

이동물체의 실시간 궤적을 추적하는 경우, 현재까지 제시된 알고리즘에는 Edge Detection, Projection method 등이 있다. 그러나 이들은 추적을 원하는 물체의 특성을 파악하는데 많은 시간을 소비한다. 그리고 콘볼루션(Convolution) 계산으로 인해 시스템의 메모리 자원을 많이 소비하며, 또한 느린 데이터 처리로 인한 실시간 비전 데이터를 놓치는 경우가 많다.

그래서 본 논문에서는 움직이는 물체에 대한 효율적인 궤적 추적 알고리즘을 제시한다. 먼저 움직이는 물체의 실시간 궤적 추적 시스템에 적용될 때 나타나는 제약조건을 언급하겠다. 첫째로 데이터의 샘플링 주파수를 높이는 것이 관건이다. 이러한 샘플링 주파수는 움직이는 궤적추적의 품질과 물체를 포착하지 못하는 에러를 유발하는 원인이 된다. 둘째로 방대한 비전 데이터의 처리량이다. 이는 많은 계산량으로 인한 부하를 야기하고 그로 인해 시스템의 성능을 저하시킨다. 셋째로 주위의 다양한 잡음 요소(광량,그늘)에 민감하다. 결국 패턴인식에 있어서 알고리즘의 문제이지만 여기서는 빛과 같은 주위의 여건에 너무 민감한 반응을 보이면 같은 화면을 샘플을 하여도 주위의 미세한 광량에 따라 판이한 데이터를 추출할 수도 있기 때문에 이런 잡음성분에 강하고 우리가 원하는 물체의 변화량에는 민감할 수 있는 요구를 만족시키도록 해야 한다.

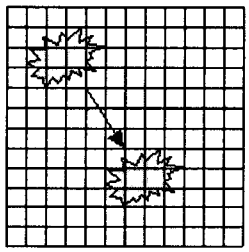
위의 조건들이 가지는 상관관계와 물체궤적을 추출하는 최적의 해석방법을 이 글을 통해서 제시하고자 한다.데이터 전송 방식에 대하여 연구한다.

2. 연구배경

이전까지의 알고리즘에서는 정확한 물체의 경계

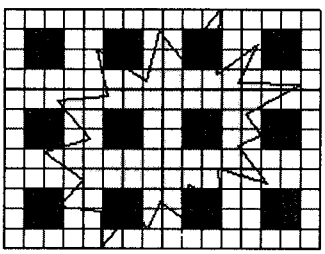
를 찾아서 정확한 물체의 인식에 초점을 둔 나머지 많은 연산량으로 인한 성능저하가 많았다. 그래서 실시간 물체의 추적을 구현에도 많은 난점이 있었다. 그러나 본 논문에서는 물체의 경계 부분을 찾는 것이 아닌 물체의 이동 벡터 성분에만 관심을 가지고 있다.

이동물체 검출을 위한 제안하는 알고리즘은 다음과 같이 간략히 된다. 먼저 입력영상에서 물체의 이동은 다음과 같다.



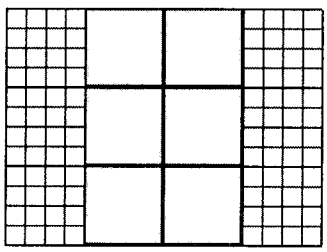
[그림1] 입력 영상에서 물체의 이동

제안하는 알고리즘은 물체의 인식을 다음의 그림 2와 같이 단순화하는 것이 우선된다.



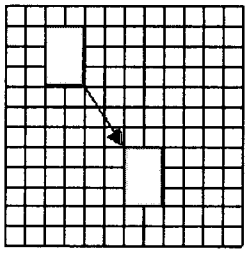
[그림2] 입력 영상에서의 이동 물체

본문에서 설명될 알고리즘에 의해 이동물체를 단순화 하면 그림 3과 같이 단순화할 수 있다.



[그림3] 단순화된 이동 물체

단순화된 이동물체를 다음과 같이 구현함으로써 연산량을 대폭 감소시킬 수 있어 실시간의 연산이 가능하다.



[그림4] 단순화된 물체의 이동

위의 그림에서 그림 1과 같이 표현된 물체의 이동 벡터성분과 그림 4와 같이 표현된 물체의 벡터성분은 동일하다. 이때, 물체의 단순화를 통한 벡터 성분의 추출이 관건이 된다. 따라서, 그림 2와 같이 회색 사각형 부분으로 샘플링하면, 블록의 배경화면과 물체를 구별이 가능하다. 이를 전체적으로 표현하면 그림 4와 같이 표현된다. 이때 회색 사각형 블록을 대표 블록(Representative Block)이라 정의하고 격자모양의 배경을 Matrix필터라 제안한다.

3. 대표 블록(Representative Block)

화상 데이터를 하나의 픽셀(Pixel)단위로 화상처리를 하게 되면 화상 데이터 연산량이 늘어날 뿐 아니라 데이터량도 커지게 된다. 이에 데이터량을 줄이면서 화상 데이터 처리의 속도를 개선하기 위해서 화상을 일정크기로 나누는 대표 블록을 정의한다.

$$A_{m,n} = F_{m,n}(Original\_image\_data) \qquad (1)$$

식 (1)은 전체 화상 데이터 행렬을 부분 블록 행렬로의 매핑(Mapping)을 나타내며, 정의된 대표 블록의 개념을 이용하여 원 화상 데이터를 블록화하는 과정을 나타내고 있다.

$$Ts = \frac{T}{number\_of\_Frame} \quad (2)$$

식 (2)에서 샘플링 시간(Ts)은 일정시간(T) 처리할 수 있는 프레임의 개수와는 반비례한다. 샘플링 시간을 줄이는 방법으로 공간의 대표 블록화를 통해서 샘플링 시간을 줄인다

$$H \times V = Frame\_Size \quad (3)$$

$$\frac{(H \times V)}{(M \times N)} = \frac{Frame\_Size}{M \times N} \quad (4)$$

H : 화상 데이터의 수평 픽셀 수

V : 화상 데이터의 수직 픽셀 수

M : 수직 대표블록 수

N : 수평 대표블록 수

$Frame\_Size$: 화상 데이터의 크기

위에서 대표 블록화를 통하여 $\frac{1}{M \times N}$ 로 다운 샘플링 한다. 표 1에서 언급된 것과 같이 연산량을 원 데이터의 크기가 10000이라고 가정할 때 100×100 으로 처리되지 않고 10×10로 줄어들어 데이터량을 감소시키는 효과를 가져온다.

아래의 표 1은 대표 블록화에 대한 행렬크기의 비교를 나타내었다.

표1.대표블록의 비율에 따른 행렬 크기의 비교

화상 데이터	M	N	비율	감소된 데이터량	행렬의 크기
100×100	1	1	1	0	100×100
100×100	10	10	1/100	9900	10×10
100×100	4	4	1/16	9375	25×25

4. 행렬필터 (Matrix Filter)

앞의 대표 블록화에서 보았듯이 상당량의 잡음이 화상 데이터에 실려있다. 이를 처리하기 위해서는 저역 통과 여과기(Low Pass Filter)가 있어야 한다.

$$g(x,y) = \left(\sum_{i=1}^{N} \sum_{j=1}^{N} a_{i,j} f(x + i - \frac{N+1}{2}, y + j - \frac{N+1}{2}) \right) \quad (5)$$

이때의 $f_{i,j}$ 는 입력된 화상 데이터이고, $g_{i,j}$ 는 저역통과 여파기를 거친 디지털 화상 데이터이며, $a_{i,j}$ 는 식 (6)과 같이 주어진다.

$$a(N,N) = \begin{pmatrix} 1 & 1 & \bullet & 1 & 1 \\ \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet \\ 1 & 1 & \bullet & 1 & 1 \end{pmatrix} \quad (6)$$

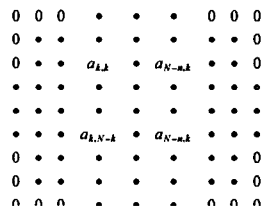
식 (5)의 저역 통과 여과기(Low Pass Filter)를 거치면 고주파 잡음 성분은 줄이는 효과는 있지만 계산량이 증가된다.

$$\frac{\left(\sum_{i=k}^{N-k} \sum_{j=k}^{N-k} a_{i,j} f_{i,j} \right)}{n^2} = \frac{\hat{Sum}}{n^2} \quad (7)$$

$$n = 2 \times k \quad (8)$$

$f_{i,j}$: 입력된 화상 데이터의 각 원소를 나타낸다.

따라서, 기존의 저역 통과 필터(Low Pass Filter)는 방대한 양의 화상 데이터를 실시간으로 처리하기에는 불가능하다. 그래서 방대한 양의 화상 데이터를 처리하기 위해서 새로운 형태로 화상 데이터를 식 (7)에 따라 모델링하고, 이를 행렬필터로 정의한다. 이를 나타내면 아래와 같다.



[그림5] 제안된 Matrix 필터의 형태

$$\sum_{j=1}^{n}\sum_{i=1}^{n} a_{i,j} = Sum \quad (9)$$

$$a_{i,j} = image\_data_{i,j} + \theta \quad (10)$$

5. 실험

모델링이 된 Matrix 필터의 적합성은 잡음의 특성을 모델링을 하는 것과 동일하다. 따라서, 잡음 모델링을 검증하기 위해 다음의 두가지 경우가 고려되어야 한다.

첫째, 부분적인 광량의 변화에 따른 물체 궤적 추적의 안정성을 하면 다음과 같다.

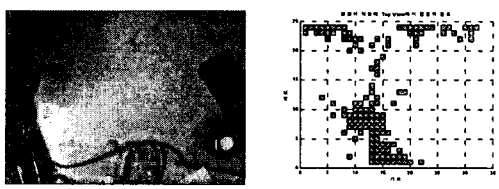
(a) (b)

[그림6] 잡음 특성을 모델링하기 위한 화상

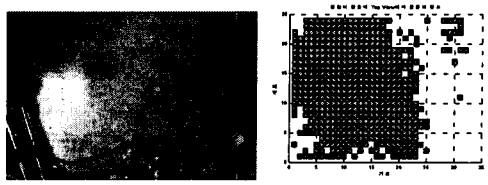
(a)와 (b) 는 동일 조건 하에서 0.1 초의 시간 간격을 두고 얻은 화상 데이터이다

그림4의 (a) 와 (b)의 화상 데이터 차분값과 신호크기를 비교해 보면 전반적으로 잡음의 크기가 신호보다 1/10보다 작다는 것을 알 수 있다. 따라서, 잡음의 특성이 신호 크기에 비해 1/10보다 항상 작다고 가정할 때, 광량 잡음성분을 변화시킬 경우와 인위적으로 화상 데이터 전체에 대해 잡음을 첨가할 경우를 비교할 필요가 있다.

둘째, 인위적인 잡음을 첨가한 경우의 안정성을 비교하면 다음과 같다.



[그림7] 광량이 적은 경우 대표블록의 분포



[그림8] 광량이 많은 경우 대표블록의 분포

이때, 광량의 증감으로 원하는 신호의 증감 뿐만 아니라 잡음의 증감도 나타날 수 있음을 알 수 있다. 따라서, 광량은 신호와 잡음 전체의 평균을 올릴 뿐만 아니라 그 크기도 증폭시킨다. 광량에 따른 잡음의 증가를 알 수 있고 그리고 그 편차의 절대값도 커짐을 알 수 있다.

6. 결론

제안된 알고리즘은 필터구조의 단순함으로 인해서 구현이 용이하며, 대표블록(Representative Block)을 가변적으로 운용할 경우에는 구간별로 대표블록의 크기를 다르게 할 수 있다.

따라서, 하드웨어 성능에 크게 의존하지 않아도 되므로, 기존의 하드웨어로도 나은 효율을 기대할 수 있다. 향후 이를 개량적으로 비교하기 위한 실험이 필요하다.

[참고문헌]

[1] T. J. Broida and R. Chellappa, "Estimation ofobject motion parameters from noisy images",*IEEE Trans, Pattern Analysis and Machine Intelligence,* vol.8, no. 1, pp.90-99, January 1986.

[2] J. L. Crowley and R. Stern, "Fast computation of the difference of low-pass transform", *IEEE Trans, Pattern Analysis and Machine Intelligence,* vol.6, pp.212-222, March 1984

[5] 김완철, 이호준, 윤경식, 최승욱, 이장명, " 행렬필터를 사용한 실시간 이동물체 추적 알고리즘" ,*대한기계학회부산지부*,pp. 194-198, April 2002.