윈도우 기반의 실시간 정지 백그라운드 모델링과 오브젝트 추출에 관한 연구

박준훈*, 최창규*, 조정현**, 김승호*
*경북대학교 컴퓨터공학과
** 영남이공대학 컴퓨터정보기술계열
e-mail : junhuni@hotmail.com

A Study on Window Based Real-Time Static Background Modeling and Object Extraction

Jun-Hun Park*, Chang-Gyu Choi*, Jeong-Hyun Cho** , Sung-Ho Kim*
*Dept. of Computer Engineering, KyungPook National University
**Division of Computer Information Technology, Yeungnam College of Science & Technology

요 약

본 논문에서는 실시간 감시 시스템의 응용분야를 위한 백그라운드 모델링과 업데이트 그리고 오브젝트 추출 시스템을 설계 구현한다. 일반적인 감시 시스템은 백그라운드의 모델링(background modeling)과 오브젝트의 검출(object detection), 오브젝트의 추적(tracking)으로 구성된다. 실시간 감시 시스템을 가능하게 하기 위해서는 작은 시간 복잡도(low time complexity)로 백그라운드와 오브젝트를 검출할 수 있어야 하고 실외환경(outdoor)의 노이즈(noise)를 반영할 수 있어야 한다. 기존에는 빠른 백그라운드 모델링을 위해 분산, 평균, 최빈값 등을 사용한 연구들이 있었다. 이러한 방법들은 빠른 수행 속도를 보장하지만 노이즈를 오브젝트로 검출하는 문제점이 있다. 또 다른 연구 분야인 메디안(median) 검출 방법은 실외환경에 존재하는 노이즈 반영에 적합한 반면, 정렬(sorting) 연산에 많은 시간이 소요된다. 본 논문은 윈도우(Window) 기반의 러닝 윈도우 리스트(Running Window List)를 이용하여 메디안 정렬 시간을 최소화하고 실시간으로 백그라운드 모델링, 오브젝트 검출, 백그라운드 업데이트를 할 수 있는 방법을 제안한다.

1. 서론

최근 컬러 디지털 캠코더가 보편화되고, PC 의 성능이 향상되면서 실시간 감시 시스템에 대한 연구가 활발히 진행되고 있다[1-7]. 영상 감시 시스템에서는 장면으로부터 움직이는 오브젝트를 분리하기 위해서 정지 백그라운드를 추출하는 방법이 연구되었다[3,5]. 지금까지 연구된 정지 백그라운드 모델링(Static Background Modeling) 방법에는 DBM, Kalman Filter, Exp. Smoothing, Median, Wallflower, Gaussian Mixture, Nonparametric model, Adaptive smoothness 등이 있다[1]. 실외환경(outdoor)에서는 바람에 의한 나무의 움직임, 물체의 반사광과 같은 노이즈(noise)가 존재하며, 이 노이즈가 오브젝트로 검출될 수 있다. 이러한 문제를 해결하기 위해서 W4 시스템[7]에서는 메디안(median) 방법을 사용한다.

메디안 방법은 N 프레임동안 정지 백그라운드를 초기화하며 이 때 사람이 전체 프레임 중 반 이상을 차지하다 사라지면 사람을 백그라운드로 인식하는 첫번째 문제가 발생한다[1]. 감시(surveillance) 시스템에서는 백그라운드의 초기화 문제뿐만 아니라 오브젝트 검출(detection)과 추적(tracking)에 관련된 많은 연산이 필요하다[5]. 일반적인 메디안 검출 방법에서는 정렬 연산의 시간 복잡도가 높기 때문에 실시간 감시 시스템에 부적합하다는 두 번째 문제점이 있다.

또 다른 문제점은 백그라운드 모델링의 N 프레임 이후의 업데이트(update) 시점이다. 업데이트 방법에는 실시간 적용 방법[2,6]과 일정 시간 단위 적용 방법[5,7]이 있다. 두 가지 업데이트 방법을 사용하기 위해서는 구름에 의한 급격한 밝기 변화와 물체의 움직임에 의한 반사광과 같은 노이즈 변화를 반영할 수 있어야 한다.

제안된 백그라운드 모델링 방법은 입력되는 RGB 컬러 영상을 HSV 좌표계로 변환하여 V(밝기), H(색상), S(채도)를 W(윈도우 크기)로 양자화 한다. 초기화 프레임 N 동안 동일한 양자화 값이 발생하면 해당윈도우에 카운트를 증가한다. 카운트 정보 중에서 크기가 1 이상인 것은 연결 리스트(linked list) 형태로 재구성하여 메디안 추출시간을 단축할 수 있다. 즉 최악의 경우 연결 리스트의 평균 크기 M 만큼 비교 연산을 수행한다. 메디안 방법의 속도 문제는 적절한 크기의 W 를 제공함으로써 해결한다. 잘못된 백그라운드 인식 문제는 오브젝트로 인식된 픽셀의 카운트가 N/2 이상이면 해당 카운트의 윈도우를 제거하고 메디안을 재검출하는 방법으로 해결한다.

초기화 단계에 사용된 N 이후의 프레임에 대하여 매 프레임 마다 해당 윈도우의 카운트를 감소하고 증가하는 방법을 사용한다. 본 논문에서는 실시간으로 백그라운드의 변화를 반영하기 때문에 업데이트 시점의 문제를 해결할 수 있다. 또한 생성된 정지 백그라운드와 N 프레임 이후의 입력 영상의 차를 이용하여 오브젝트의 추출이 가능하다. 결과적으로 실시간 정지 백그라운드의 모델링과 업데이트가 가능하고, 효율적인 오브젝트 검출을 할 수 있다.

2. 정지 백그라운드와 오브젝트 추출

2.1 HSV 색상 모델

기존의 백그라운드 모델링 방법은 대부분 밝기 정보를 이용하여 정지 백그라운드를 초기화 한다 [6,7]. 밝기 정보에서는 바람에 의한 나무의 움직임과 오브젝트의 그림자는 오브젝트로 추출된다. 나무의 작은 움직임과 그림자 부분의 색상 정보는 크게 변화가 없으며, 특히 그림자의 경우 색상 정보는 변화가 작고 채도 정보는 원래 배경 보다 값이 커진다. 색상 정보와 채도 정보를 이용하여 실외환경의 노이즈를 부분적으로 제거할 수 있다[3]. 본 논문에서는 색상, 채도, 밝기 정보를 이용하기 위해 HSV 좌표계를 사용하며 수식(1)은 RGB 좌표계를 HSV 좌표계로 변환시키는 방법이다.

$$H=\cos^{-1}\left[\frac{\frac{1}{2}\{(R-G)+(R-B)\}}{\sqrt{(R-G)^2+(R-B)(G-B)}}\right],\ S=1-\frac{3}{(R+G+B)}[\min(R,G,B)],\ V=\frac{1}{3}(R+G+B) \quad (1)$$

2.2 윈도우 기반의 메디안

입력되는 RGB 컬러 영상을 HSV 좌표계로 변환하면, H, S, V 는 각각 0~360, 0~1, 0~1 의 값으로 나타난다. 실제 영상에 적용하기 위해서는 H, S, V 는 0~255 값으로 재구성되어야 하며 재구성된 H, S, V 의 값은 윈도우 크기 W 로 양자화한다. 입력 영상에서 한 픽셀은 H, S, V 각각에 해당하는 256/W 크기의 카운트 리스트(count list)를 가지며 0 으로 초기화된다. 알고리즘 1 은 순차적으로 입력된 프레임에서 픽셀 단위로 H, S, V 의 양자화 값을 계산하여 해당 카운트 리스트의 값을 1 단위로 증가한다. 카운트 리스트는 양자화된 값이기 때문에 모든 값이 변화하지는 않는다. 소수의 윈도우(window)는 0 보다 큰 값이지만 나머지 윈도우는 0 의 값으로 존재한다. 메디안 검출에는 0 보다 큰 카운트 윈도우만 사용된다. 즉 각 픽셀(pixel) 마다 0 보다 큰 카운트 윈도우로 구성된 러닝 윈도우 리스트(running window list)를 생성하면 메디안 검출과 정지 백그라운드 업데이트의 계산 시간을 단축할 수 있다. 최종적인 카운트 리스트와 러닝 윈도우 리스트의 형태는 그림 1 과 같다.

```
For( N=0; N< Init_Frame_Size; N++ ) {
    For ( Y=0; Y < Image_Height ; Y++ ) {
        For( X=0; X < Image_Width; X++ ) {
            V = Get_V( X, Y, N );                    //N 번째 프레임의 (X,Y) 픽셀.
            Quantize_V = V / V_Quantize_Size;  // V 값의 양자화.
            Count_List[X][Y][Quantize_V] ++  // 카운트 증가.
        }
    }
}                                                    //H, S 모두 V와 동일 하게 적용된다.
```

알고리즘 1. 카운트 리스트 생성

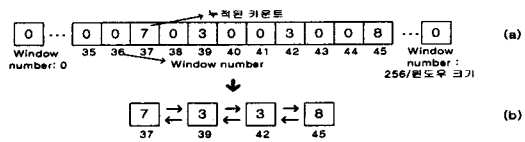

그림 1. 카운트 리스트(a)와 러닝 윈도우 리스트(b)

2.3 정지 백그라운드 추출 시점과 오브젝트 추출

메디안 정보로 이루어진 정지 백그라운드를 추출하기 위해서는 초기화 프레임 개수 N 의 정의가 필요하다. 고속도로와 같이 오브젝트가 빨리 움직이고 주변 환경의 노이즈 변화가 거의 없는 환경에서는 스코어 맵(score map)[6] 과 같은 방법을 통하여 백그라운드 검출 시점을 정의할 수 있다. 하지만, 사람과 같이 움직임이 일정하지 않은 오브젝트가 존재하는 환경에서는 맵 정보가 유용하지 않다. 즉 단일 맵 정보를 사용할 경우 업데이트 시점을 발견하지 못하는 문제가 발생한다. 기존의 방법은 초기화 시점이 불명확한 경우 실험적 결과로써 N 은 80 으로 정의한다[5].

제안된 방법에서는 검출시점을 결정하기 위해서 그림 2 와 같이 입력 영상을 10x10 크기의 SS(square split) 윈도우로 분할한다. 각각의 SS 윈도우에 속하는 픽셀의 윈도우 카운트 중에서 N/2 이상의 크기가 존재하면 해당 픽셀은 검출 가능 픽셀로 결정한다. 한 개의 SS 윈도우에 속하는 모든 픽셀이 검출 가능한 픽셀이면 해당 SS 윈도우는 백그라운드로 검출 가능하다. 즉 맵 상의 모든 SS 윈도우가 검출이 가능한 윈도우라면 백그라운드 검출 시점이 결정된다. 만약 80 프레임 이전에 검출시기가 결정되지 않으면 실험

적 결과로 N 의 값은 80 으로 정의된다. 정지 백그라운드 검출 시점 결정 방법은 알고리즘 2 와 같고 사용되는 정보는 밝기 값 V 이다.

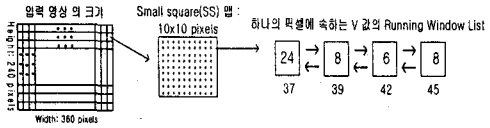

그림 2. Small Square Map

1. 하나의 픽셀에 속하는 러닝 윈도우 리스트 중에서 (N/2) + 1 보다 큰 값이 존재하면 검출 가능 픽셀로 설정.
2. SS 맵 에 속하는 모든 픽셀이 검출 가능 픽셀이면 해당 SS 맵 은 검출 가능 맵으로 설정.
3. 모든 SS 맵 이 검출 가능 맵이면 백그라운드 검출 시점.

알고리즘 2. 정지 백그라운드 검출 시점 결정

백그라운드 검출 시점이 결정되었으면 러닝 윈도우 리스트를 이용하여 H, S, V 각각의 메디안 백그라운드(Median Background) MB 를 생성한다. 각각의 MB 생성 방법은 다음과 같다. 먼저 러닝 윈도우 리스트를 순차 방문하여 카운트 값을 누적한다. 누적 값이 N/2 이상인 CP(Current Point)와 전체 누적 값 AC(Accumulated Count)를 그림 3 과 같이 생성하며 누적 값을 이용한 AC 와 CP 는 H, S, V 각각에 대하여 수행한다.

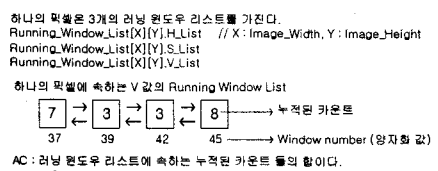

그림 3. AC, CP 생성

```
for (Y=0 ; Y < Image_Height ; Y++ ) {
  for ( X=0 ; X < Image_Width ; X++ ) {
    if ( Running_Window_List[X][Y].V_CP > (2/N)+1 ) {
        해당 CP Window 를 제거한뒤 CP 를 재검출한다.
        CP_Difference = ( Running_Window_List[X][Y].V_CP * V_Quantize_Size ) - Image[X][Y].V
        // CP_Difference 는 재검출 된 V_CP 와 실제 영상의 V 값의 차 이다.
        if ( CP_Difference > V_Quantize_Size ) {          // 재 검출된 CP 비교.
            if ( Running_Window_List[X][Y].S_CP > Image[x][Y].S  // 그림자 노이즈 검출.
              && Running_Window_List[X][Y].H_CP < H_Quantize_Size ) {
                    바람에 의한 나무의 움직임, 그림자의 움직임 과 같은 노이즈로 판단.
                    노이즈 로 판단된 픽셀은 0의 값으로 설정한다. // 노이즈 픽셀
            }
            else {
                    오브젝트 픽셀은 255 의 값으로 설정한다.    // 오브젝트 픽셀
            }
        }
        else {
                    백그라운드 픽셀은 0 의 값으로 설정한다.    // 백그라운드 픽셀
        }
    }
  }
}
```

알고리즘 3. 오브젝트 검출 과정

생성된 세 개의 MB 와 N+1 프레임의 차 영상을 통해서 오브젝트를 추출한다. 기존의 밝기 정보만 이용하는 경우 단일 임계 값을 사용한다[1,5-7]. 제안된 방법에서는 나무의 움직임, 그림자와 같은 노이즈를 부분적으로 제거하기 위해 H, S, V 의 세가지 임계 값을 사용한다. 이때 N/2 프레임 이상 존재하다 사라진 오브젝트는 잘못된 백그라운드로 인식하는 문제가 존재한다. 제안된 방법에서는 알고리즘 3 과 같은 방법으로 해결할 수 있다. 오브젝트로 인식된 픽셀의 윈도우 카운트 V 의 정보가 N/2 이상이면 이것을 제외한 나머지 카운트 리스트에서 다시 메디안을 검출한다. 재 검출된 V 의 실제 밝기 값(Window_Number *양자화 크기)과 N 프레임 이후의 입력 영상의 밝기 값의 차가 임계 값(V_Quantize_Size)보다 작으면 백그라운드로 인식하고, 크면 오브젝트로 인식한다.

2.4 정지 백그라운드 실시간 업데이트

정지 백그라운드가 생성되고 오브젝트 추출과정이 끝나면 다음 입력 프레임(N+2 프레임)을 적용하기 전에 백그라운드 업데이트 과정이 필요하다. 제안된 방법은 러닝 윈도우 리스트, CP, AC 를 이용하여 실시간 업데이트를 수행하며 크게 세가지 경우가 존재한다. 첫번째 경우는 러닝 윈도우 리스트에 업데이트 양자화 값이 존재하는 경우이다. 이때는 가장 이전의 양자화 카운트 정보를 1 감소시키고 새로운 양자화 정보를 1 증가 한다. 두 번 째 경우는 업데이트 양자화 정보가 러닝 윈도우 리스트에 존재하지 않는 경우이다. 이 때는 가장 이전의 카운트 정보를 1 감소시키고 새로운 양자화 값을 연결 리스트에 추가한다. 세 번째 경우는 마지막 카운트 정보를 감소시킬 때 러닝 윈도우 리스트가 0 값이 되는 경우이다. 이 때는 0 값이 되는 윈도우를 제거하고 새로운 양자화 윈도우를 생성한 뒤 1 증가한다. 세 개의 과정은 그림 4 와 같다.

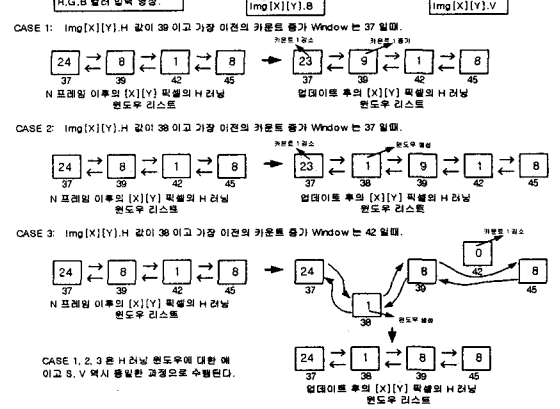

그림 4. 러닝 윈도우 리스트 업데이트

3. 실험 결과

정지 백그라운드를 초기화하기 위해서는 프레임의 크기 N 이 정의되어야 한다. 초기화 프레임의 크기는 SS 윈도우를 사용하여 결정된다. 만약 80 프레임 이내에 결정되지 않으면 실험 결과값 80 으로 설정된다.

제안된 방법에서 전체적인 시간 복잡도에 큰 영향을 미치는 양자화 크기를 수식(2)의 PSNR(Peak Signal to Noise Ratio)을 고려하여 결정한다. PSNR 은 N 프레임동안 실제 메디안과 양자화 메디안의 오차를 표현한 것이다. H, S, V 는 그림 5 의 결과에 따라 각각 색상 윈도우 크기 HW=4, 채도 윈도우 크기 SW=4, 밝기 윈도우 크기 VW=8 로 결정된다. 양자화 크기에 의하여 H 는 84 개 , S 는 84 개, V 는 32 개의 최대 윈도우 개수를 가진다. 실제 N 프레임 동안 추출된 백그라운드의 평균 윈도우 개수는 그림 6 과 같다. 즉 정지 백그라운드 실시간 업데이트에서 평균 비교 연산 횟수는 (평균 윈도우 개수/2) 이다.

$$PSNR = 10\log_{10}\left[\frac{255^2}{MSE}\right],\ MSE = \frac{1}{Height * Width}\sum_{y=1}^{Height}\sum_{x=1}^{Width}[f(x,y)-\hat{f}(x,y)]^2 \qquad (2)$$

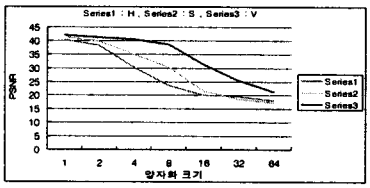

그림 5. 양자화 크기와 PSNR

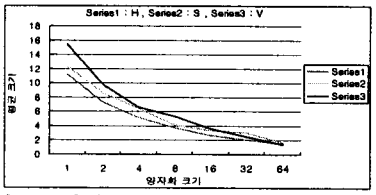

그림 6. 러닝 윈도우 리스트의 평균 크기

양자화 크기가 결정되면 N 프레임 동안 세 개의 MB 를 생성한다. 생성된 MB 와 N 프레임 이후의 입력영상의 차를 이용해서 오브젝트를 검출한다. 오브젝트 추출에 사용된 실험 영상은 자동차 표면, 창문의 유리, 나무의 움직임과 같은 실외환경의 노이즈가 존재하는 환경이다. 실험 결과 영상인 그림 7 은 79 프레임 동안 MB 를 추출하고 120 프레임에서 차 영상을 통해 오브젝트를 검출한 결과이다. 먼저 그림 7(a)영상은 120 프레임에서 실제 입력 영상이고, 그림 7(b) 영상은 노이즈 제거와 실시간 업데이트를 적용한 오브젝트 검출 결과이다. 끝으로 그림 7(c) 영상은 검출된 이진 그림 7(b) 영상을 실제 영상에 마스크로 적용하여 생성한 컬러 영상이다. 결과 영상에서 점으로 나타나는 부분은 자동차의 반사광이 노이즈 제거 단계에서 완전히 제거되지 못한 경우이다.

그림 7. 오브젝트 검출 결과

4. 결론

본 논문에서는 정지 백그라운드 생성과 오브젝트 검출을 실시간에 수행하기 위해서 윈도우 기반의 메디안 검출 방법을 제시하였다. 정지 백그라운드를 생성하기 위해서 HSV 컬러 좌표계를 이용하였으며 오브젝트 검출 시 실외환경의 노이즈를 부분적으로 제거할 수 있는 방법을 제시하였다. 제안된 오브젝트 검출 알고리즘은 H, S, V 의 값들을 이용하여 밝기 정보만 활용한 경우보다 효과적으로 실외환경의 노이즈를 제거할 수 있었다. 또한 제안한 실시간 정지 백그라운드 업데이트 모델은 평균적으로 (러닝 윈도우 리스트의 평균 크기 / 2) 의 비교연산을 수행함으로써 영상의 디스플레이(display) 지연 없이 실시간에 수행 가능하였다. 실험을 통하여 테스트한 결과 제안된 방법을 사용 시 실시간에 정지 백그라운드를 검출할 수 있고 오브젝트 추출과 실시간 정지 백그라운드 업데이트가 가능한 시스템임을 입증할 수 있었다.

참고문헌

[1] Dan Gutchess, Miroslav Trajkovic, Eric Cohen-Solal, Damian Lyons and Anil Jain, "*A Background Model Initialization Algorithm for Video Surveillance*," IEEE: Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on , Volume: 1 , 7-14 July 2001.

[2] Dashan Gao and Jie Zhou, "*Adaptive Background Estimation for Real-time Traffic Monitoring*," IEEE: Intelligent Transportation Systems, 2001. Proceedings. 2001 IEEE , 25-29 Aug. 2001.

[3] George S.K.Fung, Nelson H.C. Yung, Grantham K.H. Pang and Andrew H.S. Lai, "*Effective Moving Cast Shadow Detection for Monocular Color Image Sequences*," IEEE: Image Analysis and Processing, 2001. Proceedings. 11th International Conference on , 26-28 Sept. 2001.

[4] Ismail Haritaoglu, David Harwood and Larry S. Davis, "*A Fast Background Scene Modeling and Maintenance for Outdoor Surveillance*," IEEE: Pattern Recognition, 2000. Proceedings. 15th International Conference on , Volume: 4 , 3-7 Sept. 2000.

[5] Ismail Haritaoglu, "*W4: Real-Time Surveillance of People and Their Activities*," IEEE: Pattern Analysis and Machine Intelligence, IEEE Transactions on , Volume: 22 Issue: 8 , Aug. 2000.

[6] Andrew H. S.Lai and Nelson H. C. Yung, "*A Fast and Accurate Scoreboard Algorithm for Estimating Stationary Backgrounds in an Image Sequences*," IEEE: International Symposium on , Volume: 4 , June 1998.

[7] Ismail Haritaoglu, David Harwood and Larry S. Davis, "*W4: Who? When? Where? What? A Real Time System for Detecting and Tracking People*," IEEE: Automatic Face and Gesture Recognition, 1998. Proceedings. Third IEEE International Conference on , 14-16 April 1998.