Pan/Tilt 카메라를 이용한 목표물 자동추적장치의 구현

이의배, 성기범, 고광철
한양대학교 전기공학과

Intelligent Remote Surveillance System Using Pan/Tilt Camera

Uibae Yi, Kibum Seong, Kwangcheol Ko
Dept. of Electrical Engineering, Hanyang University

Abstract - Surveillance system on the internet has attained lots of interests recently. Computer gives surveillance system various functions like remote control and motion detecting. In this paper, auto target tracking system using Pan/Tilt camera is suggested. It consists of UNIX server and Pan/Tilt camera. When UNIX server detect motions from images it sends Pan/Tilt command to camera and camera moves by command. After finishing movement camera replies to server and server starts detecting motion again. To improve performance of motion detecting and tracking images are divided into 9 sub-regions and camera behaves differently. It is certain that robust tracking is achieved when sub-region is applied.

1. 서 론

인터넷 속도가 빨라짐에 따라 기존의 폐쇠 회로 카메라를 인터넷에 적용시키는 사례가 늘어나고 있다. 기존의 폐쇠 회로 시스템은 케이블과 코덱의 설치에 드는 비용과 번거로움, 지정된 위치에서만 모니터링을 할 수 있다는 점이 단점으로 지적 되어왔다. 인터넷을 이용한 시스템은 여기에 비해 별도의 케이블이 필요 없고, 전 세계 어디에서나 인터넷 연결이 가능한 곳에서는 모니터링 할 수 있으며, 동시 접속, 동시 제어 등이 가능하다. 본 시스템은 NetBSD(Unix) 운영체제를 이용하는 서버와 Pan/Tilt가 가능한 카메라를 이용하여 별도의 감지기 없이, 움직이는 물체를 감지하여 추적하는 시스템을 소개한다.

2. 본 론

2.1 목표물 자동추적장치의 구성

본 연구에서 제안된 '목표물 자동추적장치'는 그림 1과 같이 구성되며 주요 구성 장치는 컨트롤러와 서버의 역할을 하는 'mmEye'와 캐논 'VC-C3' 카메라로 이루어진다. 제어 프로그램은 'gcc' 컴파일러로 C언어로 짜여졌으며, 인터넷과 연결되는 CGI 역시 C언어로 짜여졌다.

2.1.1 mmEye

'mmEye'는 본 장치에서 카메라를 제어하는 역할과 카메라에서 받은 비디오 신호를 JPEG영상으로 바꾸는 역할, 그리고 이 영상을 웹으로 보내고 인터넷상에 있는 사용자의 제어를 받아들이는 서버의 역할을 한다. CPU로는 히타치 'SH-3'를 사용하며 운영체제는 NetBSD(Unix)를 이용한다. NetBSD는 BSD4.4에 기반을 두고 있으며 여러 종류의 CPU에 이식하기 쉽도록 개발된 버전이다. 'mmEye'는 그림 2와 같이 PCMCIA 네트워 카드와 RS-232C 포트를 이용하여 인터넷과 카메라제어를 하게 된다.

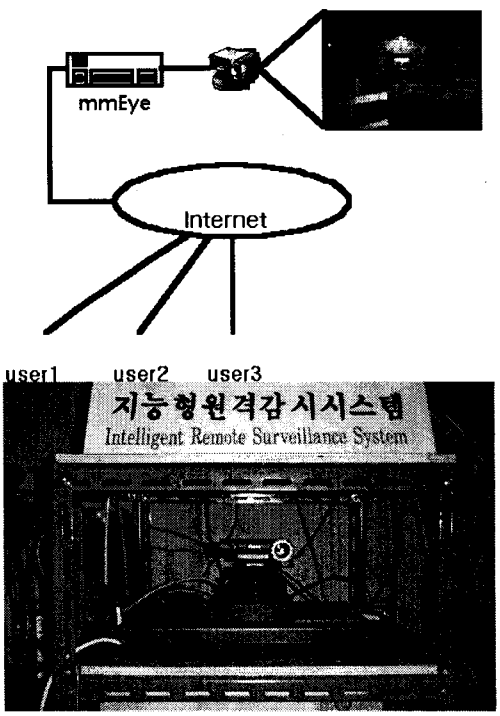

그림 1.시스템의 구성도와 사진
Fig 1. Diagram and Picture of System

본 시스템은 1년 동안 운용한 결과 한 번의 다운도 일어나지 않아 신뢰성이 확인되었다.

2.1.2 Application

UNIX 서버에서는 프로그램이 실행되는 동안 현재 프레임과 이전 프레임을 검사하여, 움직임이 감지되면 카메라에 움직임이 일어난 화소를 중앙에 위치하도록 명령을 내린다(그림 2). 카메라는 전달받은 명령을 수행하고 그 결과를 서버에 응답한다. 응답을 받은 서버는 다시 움직임을 감시하고 이 과정을 반복 수행하게 된다.

2.2 이론

'mmEye'에서 만들어지는 이미지 파일은 YUV포맷과 JPEG포맷이 있는데, 본 장치에서는 이미지 처리 속도 등과 프로그램의 작성과 실행에 있어서 간결한 YUV포맷을 이용한다. 그리고 YUV포맷에서는 이미지의 정보중

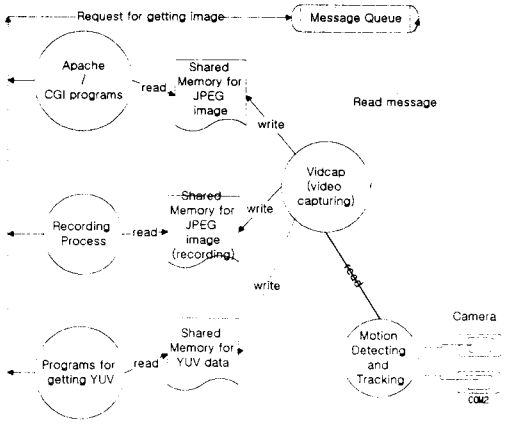

그림 2. 작업 구성도
Fig 2. Process Configuration

대부분이 Y성분에 저장되므로, Y(luminance)로 256 계조 의 320 * 240 그레이 스케일 이미지를 구성한다. 'mmEye'의 YUV 포맷은 YUYVYUYVYUYV... 순서로 저장되므로 2바이트씩 떼어서 저장하면 Y성분만의 이미지가 얻어진다. 화면의 변화를 감지하기 위해서 이 이미지에 Difference Pictures, 사이즈 필터, superpixels, likelihood ratio 알고리즘을 적용한다.

2.2.1 움직임 감지

프레임 $F(x,y,j)$와 $F(x,y,k)$간의 Difference Picture $DP_{jk}(x,y)$는 다음과 같이 얻어진다.

$$DP_{jk}(x,y)=\begin{cases}1 & \text{if } |F(x,y,j)-F(x,y,k)| > \tau \\ 0 & otherwise\end{cases} \quad (1)$$

여기서 τ는 문턱치 값이다.

또한 잡음 화소에 대한 방비책으로 사이즈 필터를 이미지에 적용한다. 변화된 화소가 무리에 속해있지 않으면 잡음화소로 처리하여 0, 즉 변화가 없는 화소로 간주된다. 사이즈 필터에는 4방향과 8방향이 있는데 본 장치에서는 8방향 필터를 이용하였다.

이 방법을 더욱 신뢰성 있게 만들기 위해 superpixels와 likelihood ratio 기법을 적용하였다. 320 * 240 화소의 영상은 64 * 48 화소의 영상이 된다. 그리고 Difference Pictures는 다시 다음과 같이 정의된다.

$$DP_{jk}(x,y)=\begin{cases}1 & \text{if } \lambda > \tau \\ 0 & otherwise\end{cases} \quad (2)$$

$$\lambda=-\frac{\left[\frac{\sigma_1+\sigma_2}{2}+\left(\frac{\mu_1-\mu_2}{2}\right)^2\right]^2}{\sigma_1*\sigma_2} \quad (3)$$

여기서 μ와 σ는 평균 명암도와 편차를 나타낸다.

그림 3과 같이 64 * 48의 Difference Pictures에서 8방향 사이즈 필터를 사용하여 검출된 여러 개의 화소군들 중 가장 크기가 큰 화소군의 중심점을 움직임이 일어난 superpixel로 정의한다.

2.2.2 움직임 추적

움직임을 감지하고 추적하는 시스템은 상당히 많은 양의 계산이 필요하고, 다른 디지털 영상 처리와는 달리 카메라와의 실시간 동기화가 요구된다.

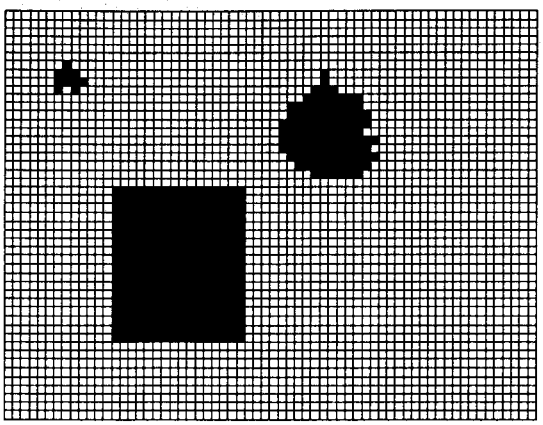

그림 3. 움직임 감시
Fig 3. Motion Detecting

한 번의 감지와 추적에 요구되는 시간 T는 다음과 같다. 이 작업은 계속 반복되게 된다.

$T = T_d + T_t$
T_d = 이전 프레임과 현재 프레임을 비교하여 카메라에 Pan/Tilt 명령을 내리기까지의 시간
T_t = 명령을 받은 카메라가 명령을 수행하고 프로그램에 응답하기까지의 시간

T_t를 줄이기 위해서는 잦은 카메라의 움직임을 억제할 필요가 있다. 따라서 추적속도의 향상과 신뢰성을 위해 영상을 그림과 같이 9개의 부영역으로 나누었다.

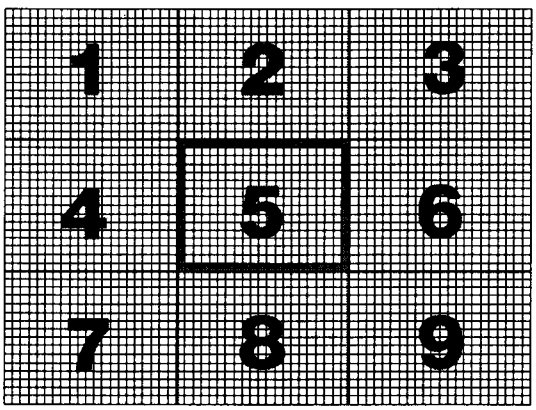

그림 4. 9개의 영역으로 나누어진 화상
Fig 4. Image divided into 9 sub_region

위의 그림에서 영역 5에서 움직임이 감지되었을 경우에는 카메라는 움직이지 않는다. 그리고 영역 2, 8에서는 감지된 화소가 영역 5의 회색 테두리중 윗줄이나 아랫줄에 위치하도록 수직 방향으로만 카메라가 움직이며, 4, 6에서는 화소가 왼쪽이나 오른쪽 테두리에 위치하도록 수평 방향으로만 움직이게 된다. 마지막으로 1, 3, 7, 9 영역에서는 화소에서 가장 가까운 테두리인 왼쪽 위, 오른쪽 위, 왼쪽 아래, 오른쪽 아래로 카메라가 수평, 수직으로 움직인다. 따라서 움직임이 일어난 화소는 영역 5에서 벗어나지 못하도록 한다. 본 방법은 카메라의 움직임

을 최소화함으로써 T_t를 최소화하고 T_d에 여유를 주어 움직임을 놓치는 경우를 방지한다. 또한 감시 주기가 짧아져 움직임 감지에 신뢰성을 높인다.

3. 결 과

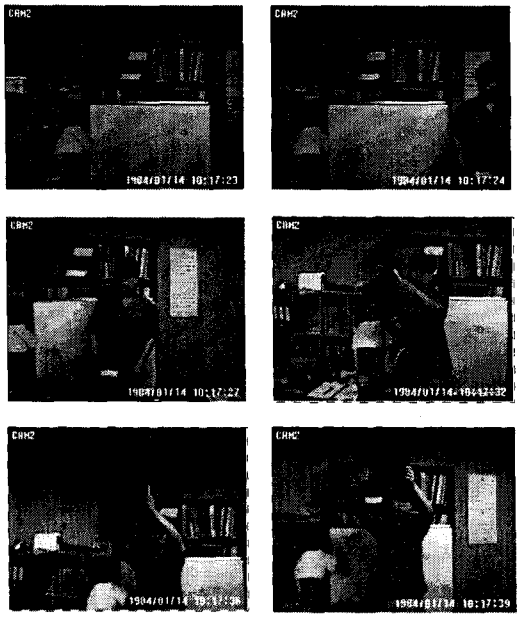

그림 5. 자동추적장치의 캡쳐 화면
Fig 5. Captured images

그림 4는 23초부터 39초까지 목표물 자동추적한 것을 1초 마다 캡쳐하여 그 중 특징적인 6 프레임을 모아 논 것이다. 첫 번째 그림에서 오른쪽에 사람이 나타나고 있고 두 번째와 세 번째에서 그 사람을 추적하였다. 네 번째에서 여섯 번째까지의 그림은 사람이 들고 있는 물체를 카메라가 추적하는 모습이다. 9개의 부영역을 이용함으로서 반응시간의 단축과 그로 인해 시스템의 신뢰성을 높일 수 있었다. 위의 결과와 같이 급격한 움직임이 아닌 경우에는 추적 시스템이 신뢰할 수 있는 반응을 보이는 것을 알 수 있다.

4. 결 론

본 논문에서는 UNIX 서버와 Pan/Tilt 카메라를 이용하여 목표물 자동추적장치를 구현하였다. Difference Pictures 기법 등을 움직임 감지에 사용하였으며, 시스템의 속도와 신뢰성을 높이기 위해서 화면을 9개의 부영역으로 나누는 방법을 제시하였다.

제안된 시스템은 무인 점포나 현금자동 출납기, 창고 감시 등에 사용될 것으로 기대된다. 하지만 아직 달리는 사람 등에 대한 실험에서는 충분히 대응하지 못하고 있으므로 움직임 예측 등에 관한 연구가 필요하다고 생각된다.

[참 고 문 헌]

[1] H.-H. Nagel "Formation of an object concept by analysis of systematic time variations in the optically perceptible environment.", Comp. Graph and Image Proc., 7, 149-194, 1978

[2] Hieu T. Nguyen, "Detection of Moving Objects in Video Using a Robust Motion Similarity Measure",Trans. on Image Processing, 1, 137-141, 2000

[3] Ramesh Jain, "Machine Vision", McGraw Hill, 1995

[4] John Miano, "Compressed Image File Formats", Addison Wesley, 1999

[5] Brains, "mmEye Programmer's Guide & SDK", Brains, 1999