영상을 이용한 감시 시스템

김성모[0], 조지만, 진민식, 정의훈, 최진구
한국산업기술대학교 컴퓨터공학과
{ddaddus[0], tjswkson, jinryu, chjeong, jkchey}@kpu.ac.kr

Video Surveillance System

Sung-Mo Kim[0], Ji-Man Cho, Min-Sik Jin, Eui-Hoon Jeong, Jin-Ku Choi
Dept. of Computer Engineering, Korea Polytechnic University

요 약

최근 인터넷과 통신이 널리 사용되면서 보안에 대한 의식과 관심은 점차 높아지는 추세이다. 기존의 보안 프로그램은 비싼 가격과 복잡한 장비를 요구하기 때문에 대중적으로 보급되는데 한계를 나타내고 있다. 따라서 널리 보급될 수 있는 보안 프로그램이 되기 위해서는 간단한 장비와 저렴한 가격으로 보안 프로그램의 설치, 운용이 가능해야 한다. 본 논문에서는 간단한 PC 카메라를 이용해서 기존의 프레임 비교의 비효율적인 방법을 개선하여 블록단위로 나눠서 필요한 일부분만의 비교를 통해 빠르고 간단하게 영상을 감지하는 알고리즘을 구현하고 감시 시스템을 개발 하였다.

1. 서 론

점차 보안의 의식과 관심이 높아지고, 생활의 편의가 중요해 지면서, 개인 보안 기기들의 필요성이 늘어가고 있다. 하지만 보안기기들을 설치하거나 유지하는 비용이 많이 들고, 설치나 구현을 하기에도 어려움이 있다. 현재의 비디오 감시시스템들은 비디오카메라 등의 장비를 이용해야 하거나 저장용량이 너무 방대하기 때문에 데이터 저장 공간을 따로 만들어서 관리해야 하는 번거로움이 있다. 또한 가격이 비싸고 부피를 크게 차지하는 장비를 이용하지 않고, 저장용량 확보 및 관리라는 번거로운 작업을 최대한 축소시켜서 쉽게 사용 가능한 새로운 비디오 감시 시스템이 필요하다.

본 논문에서는 이런 부분들을 보완하여서 누구나 쉽게 사용할 수 있도록 개인용컴퓨터 카메라를 이용한 감시 시스템에 대한 연구를 수행하였다. 영상처리부분에서의 가장 많은 처리량을 요구되고 있으므로 영상처리부분에서 가장 중요한 알고리즘을 제안하였으며 감시시스템에 적용하였다. 본 논문의 구성은 다음과 같이 정리하였다. 2장에서는 관련연구에 대해서 서술하였고 3장에서는 본 시스템의 설계 및 기능에 대하여 서술하였다. 4장에서는 본 시스템의 구현 및 성능평가 부분을 서술하였고, 5장에서는 결론에 대하여 서술하였다.

2. 관련연구

2.1 블록 정합 알고리즘

영상을 일정한 크기의 블록(Block)으로 분할하고 블록 내의 모든 픽셀(Pixel)을 하나의 움직임 벡터로 표현하는 방법이다. 움직임 벡터를 찾기 위해 이전 프레임의 블록들을 한 픽셀씩 이동하면서 현재의 블록과 가장 유사한 블록을 찾는 것이며, 정합 기준으로는 평균 절대 오차(MAE: mean absolute error)와 평균 제곱 오차(MSE: mean squared error) 등이 많이 쓰인다. 물체의 움직임은 한 블록 내에서 모두 동일하다고 가정하는데, 블록의 크기가 작을수록 이러한 가정의 신뢰도는 높아지나 움직임 벡터의 계산량과 전송량이 증가하게 된다.

2.2 영상 획득

영상들은 숫자들의 2차원 배열로 컴퓨터에 저장된다. 이 숫자들은 컬러 또는 명암도 등급, 광도, 색차 등등의 서로 다른 정보와 대응된다. 우리가 컴퓨터로 어떤 영상을 처리하기 전에, 그 영상을 디지털 형태로 만들 필요가 있다. 그러기 위해서 디지타이저가 필요하다. 디지타이저의 두 가지 기능은 샘플링과 양자화이다. 샘플링은 하나의 영상을 표현하기 위하여 동등한 공간 크기로 데이터 포인터들을 획득한다. 이 데이터 포인터들이 컴퓨터에 저장되려면, 이진 형태로 변환되어야만 한다. 양자화는 각 값에 이진수를 할당한다.

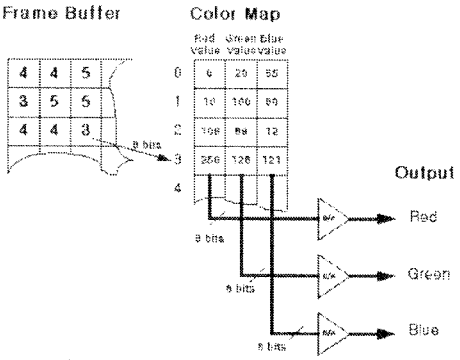
2.3 영상 저장

영상을 그림으로 저장하는 2가지 방법에는 비트맵(bitmap, raster라고도 부른다) 방법과 벡터 표현 방법이 있다. 비트맵으로 된 영상들은 그 영상들의 모든 요소를 기록한다.

하나의 원을 가진 영상은 "circle(50,50,40)"과 같은 형태의 벡터 형식으로 저장되어지고 다른 데이터 파일들과 같이 디스크에 저장된다. 영상들은 어떤 형태를 가진 헤더(header)가 있고 그 뒤에 연속해서 데이터들이 일렬로

배치된 형태로서 파일로 저장된다. 헤더는 반드시 필요하고 영상 프로세서에게 영상 정보를 제공한다. 만일 헤더가 없다면, 그 프로세서는 미리 영상에 대한 정보를 가지고 있어야 한다. 헤더는 ID 비트, 영상의 해상도, 화소당 1비트 수, 컬러 맵 데이터, 저장 기술, 영상에 대한 설명 및 정보를 포함한다. ID 비트들은 어떤 파일 형식이 사용되었고 형식 명세의 어떤 개정판을 사용했는가를 기호화할 수 있다. 몇몇 그래픽 디스플레이 시스템들은 화소당 1 또는 2 바이트의 저장 공간만을 할당할 수 있다. 그러나 이것은 화소당 3바이트를 요구하는 컬러 영상을 디스플레이 할 때 문제가 된다. 이러한 문제를 해결하는 방법은 컬러 맵을 사용하는 것이다. 컬러맵은 16.7만컬러(224)를 256(28) 또는 65,536(216) 컬러로 줄이기 위한 참조 테이블이다. 흔히 컬러 맵은 각 영상에 대한 컬러들과 그 컬러의 빈도를 분석하여 결정된다. 컬러 맵은 단순한 24비트 값의 배열이다. 그래서 각 화소들은 실제 값들이 아닌 이 배열의 요소에 대한 포인터들로 구성된다.

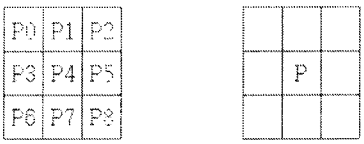
그림1은 8 비트 컬러 맵이 어떻게 동작되는지를 보여준다. 컬러 맵을 사용하는 영상 파일들은 흔히 이 컬러 맵을 영상 헤더에 저장한다.



[그림 1] 컬러 맵의 동작

2.4 Image Filtering (Median Filter Image Processing)

어떤 영상에 잡음(noise)가 있다고 할 때, 그 영상을 보고 알 수 있는 것은 잡음의 농도와 그 주변 농도의 급격한 농도차가 있다는 점과 급격한 농도차가 있기 때문에 화질이 떨어진다. 이러한 잡음의 성질을 이용하여 잡음 제거하는 수법을 smoothing이라고 부른다. 단, 영상 데이터의 에지(edge)부분도 급격한 농도차가 있기 때문에 이 에지의 부분과 잡음 부분을 어떠한 방법으로 분리하여 잡음만을 제거하는가가 smoothing 과정의 중점이 된다. 이를 위한 간단한 잡음 제거법이 이동 평균법(Mean)이다. 이것은 아래 그림과 같이 어떤 화소 주변의 3x3 화소의 평균치를 그 화소의 값과 교환하는 기법이다. 이 기법은 영상을 흐리게 하면, 세밀한 잡음은 눈에 보이지 않게 된다는 점에서 착안 되었다.



[그림 2] Mean filter에 의하여 화소 값을 구하는 방법

2.5 ODBC

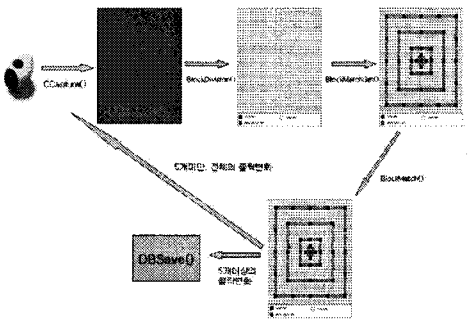
ODBC는 데이터베이스를 액세스하기 위한 표준 개방형 응용 프로그램 인터페이스이다. 프로그램 내에 ODBC 문장을 사용하면 MS-Access, dBase, DB2, Excel, Text 등 여러 가지 종류의 데이터베이스를 액세스 할 수 있다. 이를 위해서는 ODBC 소프트웨어 외에, 액세스할 각 데이터베이스마다 별도의 모듈이나 드라이버가 필요하다.

3. 시스템 구현

시스템 구현의 기본기능으로는 이미지 캡쳐기능, 이미지 재생기능, 이미지 보정기능, 달력을 이용한 예약감시기능, 사용자 로그인 지정기능을 가진다.

3.1 이미지 캡쳐와 이미지 재생기능

PC카메라로 입력받은 이미지를 블록 정합 알고리즘을 사용하여 그림3의 방법으로 움직임을 감지한다.



[그림 3] 블록 정합 알고리즘을 이용한 캡쳐 동작

그림3의 방법으로 감지된 이미지는 시간의 흐름에 따라 DB에 순차적으로 저장되고, 이 저장된 이미지를 재생하게 되면 가장 먼저 저장된 이미지부터 순서대로 재생된다.

3.2 블록 정합 알고리즘

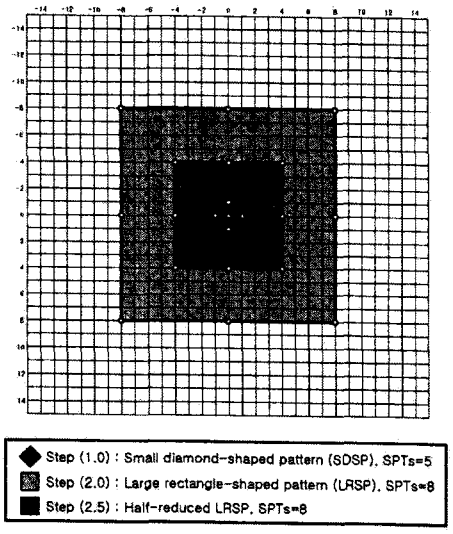
본 시스템에서 사용할 제안 알고리즘은 크게 3단계의 탐색 절차를 가진다.

(1) 작은 다이아몬드 형태의 패턴(small diamond-shaped pattern, SDSP)의 탐색점을 탐색하고,[1],[2][3] (2) 윈도우의 중앙을 중심으로 설치된 사각형 패턴(large rectangle-shaped pattern, LRSP)의 탐색점을 추가적으로 탐색한 후, (3)작은 다이아몬드 탐색으로 탐색을 완료한다. 다음은 제안알고리즘의 각 탐색 단계에 대한 세부적인 설명이다.

(1) 단계: 작은 다이아몬드 형태의 패턴(SDSP)의 탐색점를 탐색.
중앙의 작은 다이아몬드 형태의 패턴(SDSP) 위에 5개의 탐색점을 배치한다. 최소 BDM점이 탐색 윈도우의 중앙이라면 탐색을 완료하며(MV=ZMV), 그렇지 않다면 (2)단계를 진행한다.

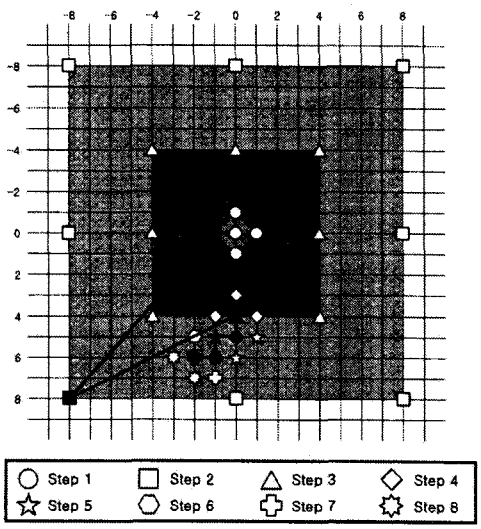
(2) 단계 : 사각형 패턴(LRSP)의 탐색점를 탐색.

탐색윈도우의 절반크기인 사각형 패턴(LRSP)를 가정한다. 사각형 패턴(LRSP)의 중심점이 탐색 윈도우의 중앙이 되도록 배치하고 그 패턴위에 놓인 8개의 탐색점들을 이전 단계의 최소 BDM점과 비교 탐색한다. (1)단계의 탐색점가 최소 BDM점이라면 (3)단계를 진행한다. 그렇지 않다면 다음의 부차적인 (2.5)단계를 진행한다. 참고로, W=7인 경우 사각형 패턴(LRSP)크기는 9×9이며, W=15인 경우 17×17이다.



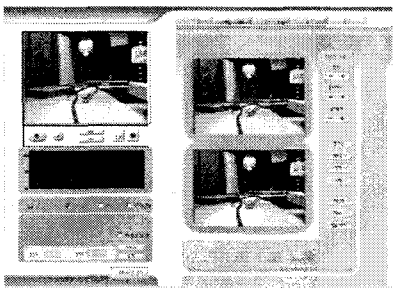
(2.5) 단계 : 사각형 패턴(LRSP)의 축소.
사각형 패턴(LRSP)의 크기를 절반으로 축소한다. 축소된 사각형 패턴(LRSP)의 크기가 9×9 미만이라면 (3)단계를 진행한다. 그렇지 않다면 사각형 패턴(LRSP)를 탐색 윈도우의 중앙에 배치하고 그 패턴위에 놓인 8개의 탐색점들을 이전 단계의 최소 BDM점과 비교 탐색한다. 그리고 (2.5)단계를 반복한다.

(3) 단계 : SDS 알고리즘의 수행
이전단계의 최소 BDM점이 3×3 크기를 가진 작은 다이아몬드 형태의 패턴(SDSP)의 중앙이 되도록 하고 작은 다이아몬드 형태의 패턴(SDSP)의 나머지 탐색점들을 탐색한다. 최소 BDM점이 작은 다이아몬드 형태의 패턴(SDSP)의 중앙이라면 탐색을 중단한다. 그렇지 않다면 (3) 단계의 과정을 반복한다.



3.3 이미지 보정기능

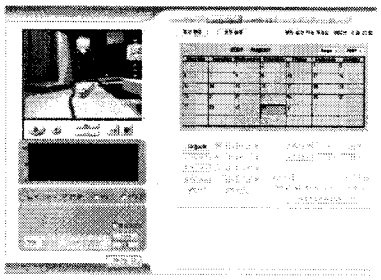
카메라로 입력받은 이미지가 주의 환경에 따라서 명암이 다르므로 보정이 필요할 때 조정가능하다. 이미지 보정 기능으로는 밝기 조절, 콘트라스트 조절, 날카로움 조절의 이미지 조절 부분과 영상을 변환 시키는 경계선 추출 필터, 반전 필터, 엠보싱 필터, 흑백 필터와 영상을 개선시키는 부분의 컬러영상 향상필터, 흑백변환 향상필터, 그리고 노이즈를 제거시키는 필터로 구성되어 있다. 실시간 영상이나 재생 부분에서 캡쳐되는 이미지들이 자동으로 로드되어 이미지를 보정하거나 사용자가 직접 이미지 파일을 불러들여 이미지를 필터링 할 수 있다.



[그림 4] 이미지 보정 기능

3.4 보안계획

사용자가 감시시스템을 지키고 있지 못하거나 자리를 비우는 경우에 예약 감시 기능을 지정할 수 있다. 보안계획에서는 달력화면이 출력되며 해당 날짜의 통계그래프가 나타난다. 그림5에서의 화면처럼 보안 계획을 결정하여 예약 할 수 있는 기능이 있으며 해당 날짜에 녹화된 탐지 구간이 출력된다. 또한 그 날에 녹화된 영상을 바로 재생 할 수 있다.



[그림 5] 보안 계획 기능

4. 구현 및 성능평가

이 절에서는 먼저 PC 카메라에서 초당 30프레임으로 영상을 받아오고, 그 영상을 블록으로 나누어서 기존의 프레임 영상과 비교를 하여 임계치 이상의 특정 범위 움직임이 감지되면 사람으로 간주하여 저장하는 과정을 시험하였다. 또한 저장된 것이 재생이 되고 이미지 필터링과 해당 스케줄에 시스템이 동작 하도록 하였으며, 움직임 감지에 대한 통계, 저장 기간 등의 기능이 작동하는지 프로그램을 구성하여 위와 같은 기능하는지 실험을 하였다.

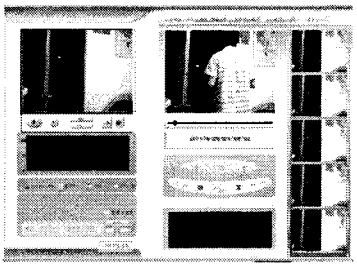
4.1 시험 환경

본 실험을 위해 기본적으로 윈도우 기반의 OS가 지원하는 컴퓨터, RAM은 512M 이상의 사양이 필요하다. PC 카메라의 모델은 윈도우 기반에서만 지원하는 Microsoft VX-3000으로써 높은 화질의 기능을 제공하는 PC 카메라를 사용하였다. 카메라가 움직임이 감지할 수 있는 거리는 7M이내이며, 카메라의 위치는 설정은 외부 환경에 잘 반응하지 않는 곳이 좋으며 문이나 중요한 위치를 감지할 수 있는 방향으로 설치하도록 해야 한다.

4.2 시험 결과

제안된 알고리즘의 탐색속도와 성능을 평가하였다. 탐색속도는 블록당 평균 탐색점 수에 따라서 속도가 완전탐색에 비교하여 37%향상이 되었다. (CIF 360 x 288, 100 프레임) 성능은 MAD로 확인하였으며 완전탐색과 비교하여 1.5% 떨어지는 것을 확인하였다. 감시시스템에서의 시험은 움직임의 감지를 위해 7M 거리에서 실험을 하였다. 움직임을 감지는 정확하게 감지되었으며 또한 영역을 설정한 후의 감지도 성공적으로 동작하였다.

움직임 감지에 대해서 재생과 이미지 필터 기능역시 가능 하였고, 부가적인 기능의 스케줄러, 재생, 이미지보정, 그래프 등의 옵션의 기능이 작동하는 것을 확인하였다.



[그림 6] 움직임 감지 메인 화면

그림6와 같이 문이 열리고 움직임이 감지가 되면 그래프의 변화와 경보가 동시에 일어나면서 날짜별로 저장이 된다. 왼쪽상단의 화면은 카메라의 감시 영역이고 가운데의 영상은 저장된 파일의 재생되는 화면이다.

5. 결론

지금까지 블록 정합 알고리즘(block matching algorithm)을 이용해서 영상 감시 시스템의 실제 구현을 통한 동작원리를 분석하여 연구의 실제 검증을 확인했다. 먼저 블록 정합 알고리즘을 제안하여 성능을 확인하였으며 타 알고리즘과의 비교하여 비슷한 수준의 성능을 확인하였다. 본 영상 감시 시스템은 저가로 구현하였으며 소규모 환경에서의 사용이 확대될 것으로 기대한다. 향후, 무선통신 등과의 연동하는 시스템으로 확장하며 영상화질과 감지 능력의 정밀도를 향상하고자 한다.

6. 참고 문헌

[1] C.H.Cheung and L.M.Po,"Novel cross-diamond-hexagonal search algorithm for fast block motion estimation," *IEEE Tranx. Multimedia.*, vol.7, no.1, pp16-22, Feb. 2005

[2] S.Zhu and K. K. Ma, "A new diamond search algorithm for fast block matching motion estimation,"*IEEE Trans. Image Process.*, vol. 9, no.2, pp. 287-290, Feb. 2000

[3] Xuan Jing and Lap-Pui Chau, "An efficient three-step search algotithm for block motion estimation," *IEEE Trans. Multimedia.*, vol.6, pp. 435-438, June. 2004