와이파이 감시 카메라기반 지능형 방범 서비스 기술

유진선*, 권일영*, 박철*, 양정호*, 박순용*
*경북대학교 컴퓨터학부
e-mail : zinus91@gmail.com

An intelligent surveillance technique based on Wifi CCTV

Jin-Sun You*, Il-Young Kwon*, Chul Park*, Jung-Ho Yang*, Soon-Yong Park*
* School of Computer Science and Engineering, Kyungpook National University

요 약

이 기술은 CCTV 의 본연의 목적인 범죄 예방을 좀 더 효과적이고 효율적으로 하기 위해 개발하였다. 먼저 사용 공간에 제약을 줄이기 위해 와이파이 기반의 CCTV 를 사용하기로 하였다. 그리고 단순히 영상을 수집하는 것이 아니라 아래와 같은 알고리즘을 영상에 적용하여 범죄 예방을 위한 판단을 좀 더 빠르고 정확하게 할 수 있도록 하였다. 첫 번째로 배경에서 객체들을 분리한 뒤 각각의 객체들을 labeling 하여 서로를 구분하였다. 그 후 특징점을 이용해 각 객체들을 추적한 뒤 객체마다 적당한 관심영역을 설정한다. 관심영역에서 색을 이용한 외부인 침입 감지기능과 객체의 속도와 방향을 이용한 폭력적인 행동 감지기능을 적용하였다. 더 나아가 사용자의 편의를 위해 서버/클라이언트 모델을 사용하여 외부인 침입이나 폭력이 감지되면 서버가 클라이언트에게 경고 메시지와 함께 현재 상태를 이미지로 보낼 수 있도록 하였고 클라이언트가 원한다면 언제든지 실시간 영상도 볼 수 있도록 하였다.

1. 서론

최근 교내에서 학원폭력 또는 성폭력과 같은 강력범죄들이 많이 일어나고 있다. 이에 대한 대책으로 학교나 정부에서는 교내에 CCTV 설치를 의무화하고 있다. 그러나 교내에 설치된 CCTV 는 대체로 비용절감을 위해 영상을 저장하지 않거나 아예 CCTV 가 꺼져있는 등 무늬만 CCTV 인 경우가 많다. 또한 CCTV 가 제대로 설치되어 있다고 해도 관리인들이 계속해서 CCTV 를 주시하지 않는다면 범죄가 일어나도 제대로 인지하지 못하는 것이 현실이다. 이러한 문제점들을 해결하기 위해 우리는 와이파이 감시카메라기반 지능형 방범 서비스를 만들기로 하였다.

언론은 최근 아동 성폭력이 일어난 원인 중 가장 결정적인 이유는 외부인이 아무 제재 없이 교내에 들어올 수 있었기 때문이라고 지적한다. 그래서 우리는 외부인과 내부인을 구분하여 관리자에게 즉시 알려줄 수만 있다면 성폭력을 예방 할 수 있다고 판단하였다. 내부인과 외부인을 구분하는 여러 가지 방법 중 우리는 옷에 주목하였다. 특정 집단에서는 여러 이유들로 인해 유니폼을 착용하곤 하는데 특히 중, 고교에서는 교복이라는 유니폼 착용을 의무화 한다. 그래서 교복을 이용한다면 충분히 내부인과 외부인을 구분할 것이다. 그 후 교복의 착용여부 판단은 색깔을 이용하기로 하였다. 왜냐하면 저마다 학교들은 각자의 개성을 나타내기 위해 교복의 모양과 색을 다르게 하기 때문이다. 그 결과 우리가 지정한 색의 옷을 입지 않은 사람이 들어올 시 그 사람은 외부인으로 판단하고 관리자에게 경보를 보낸다.

학교 폭력의 경우는 객체의 움직임을 판단하기 위해 특징점을 이용하려고 한다. 특징점이란 객체가 움직이면 함께 변하는 것으로 이 변화를 이용해 각 객체의 속도와 방향을 구할 수 있을 것이다. 만약 폭력적인 행동이 일어난다면 학생들은 평소와는 다르게 격렬하게 움직일 것이며 이때의 객체의 특징점들은 평소보다 다이나믹하게 변할 것이다. 이러한 특징점들의 변화로 인해 객체의 속도와 방향이 평소와는 다른 값을 띄게 될 것이다. 이 특정한 값을 잘 감지한다면 교내 폭력도 알아차릴 수 있을 것이다.

또한 우리 CCTV의 마지막 특징은 모바일 기기로의 전송이다. 관리자가 계속해서 CCTV를 보면서 외부인이 침입했는지 싸우는 지를 보는 것은 굉장히 지루하고 귀찮은 일이며 공간적으로도 굉장히 제한적이다. 그래서 우리는 모바일 기기를 이용해 관리자가 CCTV를 계속해서 주시하지 않아도 사건(외부인 침입, 객체간의 다툼)이 일어날 경우 바로 알아차릴 수 있도록 관리자에게 알림을 보내고, 더 나아가 관리자가 원하면 언제든지 현재의 CCTV의 영상을 볼 수 있도록 했다.

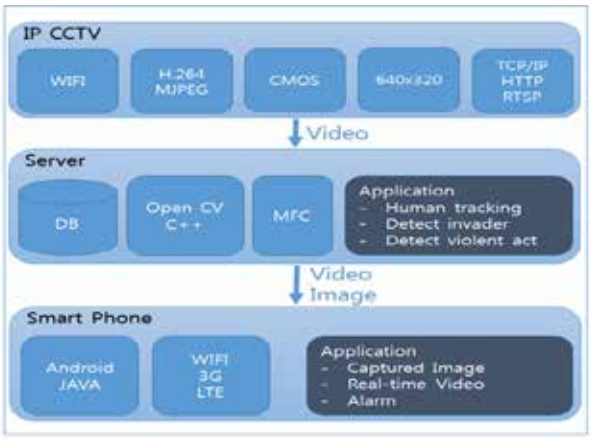
2. 지능형 방범기술 알고리즘

시스템은 공통적으로 CCTV 로 들어오는 영상을 배경과 객체로 분리한 뒤 분리된 객체에 라벨을 씌워 각각의 객체들을 구분한다. 구분된 객체들을 계속해서 추적하면서 그 후 시스템의 주요기능에 따라 두 가지로 나뉘는데 먼저 유니폼의 색을 이용해 외부인과 내부인을 구분하는 과정을 살펴보면 각 labeling 된 객체를 먼저 관심영역으로 지정한 뒤 색 정보를 받아온다. 이 때 유니폼의 색이 우리가 미리 지정한 색이

아니라면 외부인으로 판단하고 관리자의 모바일 기기로 현재의 이미지와 함께 경고를 보내면 끝이 난다.

두 번째로 속도와 방향을 이용해 격렬한 움직임을 감지하는 과정을 살펴보면 각 labeling 된 객체를 관심영역으로 지정한 후 특징점들을 추출한다. 이 특징점들은 객체가 움직일 때 마다 계속 변하게 되는데 이 때 변하는 정도를 이용해 객체의 속도와 방향을 구한다. 만약 각 객체의 속도와 방향을 이용해 구한 값이 특정 수치를 넘어가면 격렬한 움직임으로 판단하고 관리자의 모바일 기기로 현재의 이미지와 함께 경고를 보내면 끝이 난다.

2.1 시스템 아키텍처



(그림 1) 시스템 아키텍처

IP 기반 CCTV 에서는 640x480 해상도이고 H.264 파일형식으로 촬영을 한다. 촬영된 영상은 HTTP 또는 RTSP 형식으로 서버로 전송된다. 전송된 영상은 서버에서 Open CV 라이브러리를 이용해 영상을 분석하고 처리한다. 또한 MFC 를 이용해 UI 를 만들고 C 를 이용해 모바일 기기(클라이언트)와 TCP/IP 연결을 하고 경고 메시지와 함께 이미지를 전송해준다. 모바일(클라이언트)은 안드로이드 OS 를 사용하고 Java 를 이용해 UI 는 물론 서버와 TCP/IP 연결도 한다.

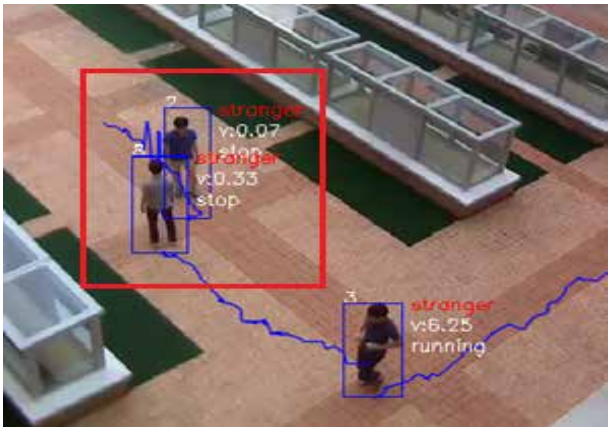
2.2 Gaussian Mixture Model 을 이용한 전경과 배경의 분리

(그림 2) 전경과 배경의 분리

배경분리에는 Gaussian Mixture Model 을 이용하였고 OpenCV 에서 제공되는 클래스를 사용하였다. 이 클래스는 기본적인 클래스로 저희 프로젝트에 사용하기에는 약간의 수정이 필요하였다. 그래서 멤버 변수의 값 중 그림자 검출, 학습 프레임 수와 threshold 값을 수정하였다. 그 결과 (그림 2)에서 볼 수 있듯이 그림자가 생성되어도 깔끔하게 제거되었으며, 50 프레임마다 배경을 계속 학습하여 중간에 배경이 변하더라도 충분히 대처할 수 있게 되었다. 또한 객체들이 분할되는 현상을 줄이기 위해 morphology 연산을 사용하여 각 픽셀을 넓어지게 하여 객체들이 가능한 한 분할되지 않게 되었다.

그러나 영상 도중 빛의 세기가 갑자기 바뀌면 Gaussian Mixture Model 로는 확실한 객체 분리가 불가능하다. 왜냐하면 제대로 객체를 분리하기 위해서는 학습기간(50 프레임)이 반드시 필요하기 때문이다. 그래서 이러한 문제는 객체 분리 이후의 과정에서 해결하기로 하였고 해결책으로 조명 정규화를 영상에 적용 시 충분히 해결이 가능하였다. 조명 정규화는 색 정보를 RGB 를 Lab 로 바꾼 뒤 L 에 Media Filter 를 적용하여 Gradation 조명의 정보를 얻어내고 이를 반전하여 반전된 조명을 얻어낸다. 이를 원본영상과 합하면 조명 정규화된 영상을 얻어낼 수 있어 급변하는 빛에도 충분히 대응할 수 있게 된다.

2.3 KLT 특징점을 이용한 객체 추적



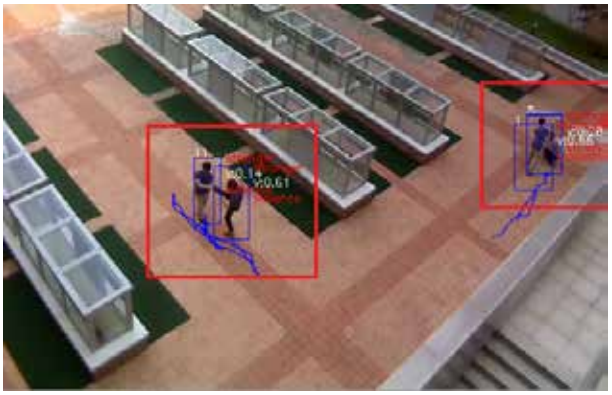
(그림 3) 객체 추적

객체추적에 앞서 먼저 각 객체들을 labeling 을 해야 한다. 왜냐하면 각 객체들을 서로 구분할 필요가 있기 때문이다. 전경과 배경을 분리하고 나면 이진화를 통해 전경은 흰색으로 배경은 검은색으로 나타난다. 이 영상에서 전경만 segmentation 을 통하여 객체를 labeling 을 한다. 그리고 인식한 객체가 새로운 객체인지 기존에 존재하던 객체인지 판별을 하게 된다. 이 때 판별기준은 객체의 좌표를 통하여 이루어 진다. 그런데 간혹 객체가 분할(상체와 하체)되는 현상이 발생하는데 이 경우에는 병합하는 과정이 필요하다. 병합 과정으로는 새로운 객체 두 개가 매우 근접해서 분포해있으면 객체가 분할 되었다고 판단한 후 두 객체를 병합한다.

이제 본격적인 추적 과정이 이루어지는데 labeling 만 이용해도 간단한 추적은 충분히 가능하다. 그러나 두 객체가 겹쳤을 시 labeling 을 하면 하나의 객체로 인식되어버리는 문제점이 발생하게 되어 올바른 추적이 불가능하다. 그래서 우리는 이 문제를 해결하기 위하여 객체에 KLT 특징점들을 부여하고 이 점들의 Optical flow 를 통하여 객체를 추적하였다. 그리고 객체가 겹쳐지면 겹쳐진 부분의 특징점들을 제외하고 겹쳐지지 않은 부분의 특징점들의 이동방향과 이동거리를 이용하여 그 다음 프레임에서 객체의 위치를 결정하였다. 이러한 방법으로 (그림 3)에서 보이듯이 2 번 객체와 3 번 객체가 겹치더라도 하나의 객체로 합쳐지지 않고 각자의 라벨을 유지하면서 끝까지 추적을 할 수 있게 되었다.

또한 각 특징점들 마다 stack 을 만들어 특징점들의 이동경로를 알 수 있도록 하였고, 대표점을 지정하여 객체의 이동경로를 화면에 표시하였다. 그리고 특징점들을 stack 에 저장함으로 인하여 단순히 객체의 좌표뿐만 아니라, 속도의 변화, 운동 방향의 변화 등의 정보를 확보하고 이를 바탕으로 객체의 여러 가지 상태를 정의 할 수 있었다.

2.4 객체의 속도와 방향을 이용한 격렬한 움직임감지



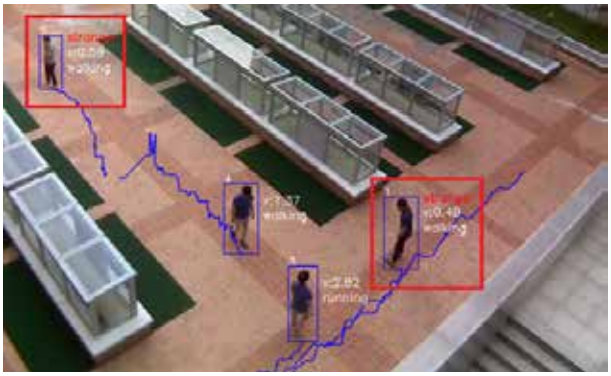
(그림 4) 움직임 판별

특징점을 이용하면 객체를 추적하면서 특징점의 변화에 따라 객체의 속도와 운동 방향을 구할 수 있다. 우리는 속도를 이용해 객체의 행동을 다음과 같은 세 가지로(stop, walking, running) 분류하였다. 객체의 속도와 운동방향의 변화 량을 기반으로 객체의 최근 50 프레임의 흔들림 정도(격렬한 움직임을 판단하는 척도 M)를 판단하였다.

$$M = \sum_{i=1}^{50} v_i \times (|\theta_i - \theta_{i-1}|)$$

이렇게 구한 M 값은 stop, walking, running 일 때는 낮게 나온다. 왜냐하면 stop 경우에는 v 가 0 에 가깝고, walking 과 running 의 경우에는 방향의 변화가 적기 때문이다. 반면 싸우는 장면에서는 매우 높게 측정된다. 왜냐하면 격렬하게 움직일 때는 방향의 변화가 잦고 v 도 0 이 될 리가 없기 때문이다. 이렇게 구한 M 값으로 (그림 4)에서와 같이 격렬한 움직임을 감지하였다.

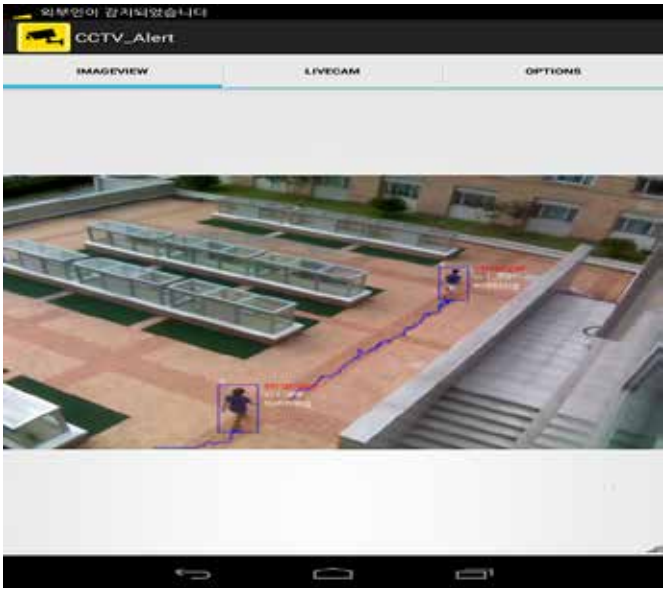
2.5 유니폼 색을 이용한 외부인 감지



(그림 4) 색 판별

프로그램의 주요 기능 중 하나인 색 판별은 미리 labeling 된 객체를 박스로 씌운 뒤 각각을 관심영역으로 지정하는 작업부터 시작한다. 각각의 관심영역으로 잡힌 객체에서 우리가 미리 지정한 색 정보(R, G, B 의 최소값과 최대값의 범위)를 넘겨 이진화를 통해 이 범위에 속하는 색이 있다면 흰색으로 출력하고 속하지 않는다면 검은색으로 출력한다. 그 후 흰색영역이 총 관심영역의 몇 퍼센트인지를 따져 유니폼의 착용여부를 따진다. 이 때, 매 프레임마다 흰색영역을 따져 침입자를 따지지 않고 약 20 프레임 동안의 흰색영역을 따진 뒤 침입자를 판단하도록 만들어 조금 더 인식률을 높였다. 우리는 RGB 의 최소, 최대값을 회색으로 지정한 뒤 각 객체의 색이 회색인지 아닌지를 판별한다. 그 결과 (그림 5)에서 살펴보면 회색 상의를 입는 객체(2 번)와 먹 색 상의를 입은 객체(1 번)를 침입자라고 판단하게 된다.

2.6 모바일 기능



(그림 6) 모바일 기능

일단 TCP/IP 소켓형식으로 서버와 클라이언트가 구성된다. 일단 서버의 경우 C 언어로 작성이 되어있으며, 클라이언트의 경우는 안드로이드 환경에서 돌아가게 되기 때문에 JAVA 및 안드로이드 SDK 를 이용해 작성하였다. 일단 프로그램이 시작되면 서버가 클라이언트의 요청을 받을 준비를 하고, 클라이언트의 요청이 있을 때까지 기다린다. 만약 클라이언트가 서버에 접속 요청을 하게 되면 TCP/IP 소켓을 통해 서버와 클라이언트가 연결된다. 서버에서는 CCTV 에서 영상을 받아와 처리를 하는 도중 외부인이 발견되면 jpg 형식으로 화면을 즉시 캡쳐하고, 해당 이미지 파일을 TCP/IP 소켓을 통해 서버에 연결된 클라이언트에게 전송한다. 해당 클라이언트는 서버로부터 이미지 파일을 받자마자 notification(스마트폰의 상단부 알림)형태로 진동음과 동시에 클라이언트 사용자에게 알리고 앱의 화면에 해당 이미지를 띄워 외부인의 침입을 알리게 된다.

또한 모바일의 또 다른 기능 중 하나로 라이브 캠 기능이 있다. 이 기능은 언제든지 현재 CCTV 상황을 실시간으로 볼 수 있도록 Webview 를 이용하여 웹 페이지를 프로그램 안에 rendering 을 하여 구현하였다. 우리가 RTSP 가 아닌 Webview 를 사용한 이유는 IP 카메라 제조사에서는 실시간 영상을 웹 페이지를 통해 제공하는 경우가 많아 쉽게 가져올 수 있기 때문이다.

3. 알고리즘을 적용한 실험 결과

3.1 유니폼 색을 이용한 외부인 감지 결과

(표 1) 유니폼 색 인식률 결과

Data set	Video1 (회색)	Video2 (회색)	Video3 (파랑색)	Vide3 (파랑색)
인식률	80.5%	82.6%	86.1%	83.9%

미리 찍어놓은 샘플 영상 4 개 중 Video1, 2 는 회색을 유니폼 색으로 정하였고 Video3, 4 는 파랑색을 유니폼 색으로 정하였을 때 유니폼 색을 이용한 외부인 감지 알고리즘을 적용한 결과 위 (표 1)과 같은 값이 나왔다. 실험 방법은 한 프레임당 색을 이용해 올바르게 판별된 객체의 수에 전체 객체의 수를 나눈 값의 전체 평균을 구하였다.

3.2 객체의 속도와 방향을 이용한 격렬한 움직임 감지 결과

(표 2) 격렬한 움직임 인식률 결과

Data set	Video1	Video2	Video3	Video3
인식률	70.8%	77.5%	83.6%	86.1%

미리 찍어놓은 격렬한 움직임을 보이는 영상 4 개를 객체의 속도와 방향을 이용하여 격렬한 움직임을 감지 알고리즘을 적용한 결과 위 (표 2)와 같은 결과가 나왔다. 실험 방법은 한 프레임당 격렬하게 움직인다고 판단된 객체의 수와 격렬한 움직임에 참여한 객체의 전체 수를 나눈 뒤 그 값의 전체 평균을 구한다.

4. 결론

먼저 유니폼 색을 이용한 외부인 감지 알고리즘의 결과를 살펴보면 평균적으로 약 82%의 인식률을 보였다. 잘나온 수치라고 생각되지만 아쉬운 부분이 있다면 멀리 있는 객체의 경우 객체의 크기가 너무 작아 확실한 색 판별이 어렵다는 점이다. 또한 RGB 의 범위를 크게 준 결과 회색과 비슷한 색 또한 회색으로 인식하여 올바른 객체가 외부인으로 감지한 경우가 있었다. 멀리 있는 객체의 경우는 어쩔 수 없는 오차라고 생각되지만 후자의 경우 RGB 의 범위를 조금만 더 정교하게 조정한다면 훨씬 좋은 결과를 얻었을 것이라 생각된다.

객체의 속도와 방향을 이용한 격렬한 움직임 감지 알고리즘의 결과를 살펴보면 평균적으로 약 80%의 인식률을 보였다. 영상에 따라 인식률의 격차가 심한 것을 볼 수 있는데 첫 번째 영상의 경우 격렬한 움직임이 단조롭고 다른 영상에 비해 격렬한 움직임이 약했기 때문이다. 이 경우 좀 더 정교하게 격렬한 움직임을 감지하기 위해 변수들을 수정한다면 걷다가 갑자기 방향을 바꾸는 행동, 조금 격하게 걷는 행동 또한 격렬한 움직임으로 판단할 수 있을 것이다.

감사의 글

본 연구는 경북대학교 임베디드소프트웨어연구센터와 컴퓨터학부의 URP 프로그램과 경북대학교 특성화사업의 지원으로 수행되었음.

참고문헌

[1] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. Proceeding of imaging Understanding Workshop, pages 121-130, 1981.

[2] Jianbo Shi and Carlo Tomasi. Good Features to Track. IEEE Conference on Computer Vision and Pattern Recognition, pages 593-600, 1994.

[3] Gary Bradski and Adrian Kaehler, "Learning OpenCV: Computer Vision with the OpenCV Library". October 1, 2008

[4] An Improved Adaptive Background Mixture Model for Realtime Tracking with Shadow Detection. 2nd European Workshop on Advanced Video Based Surveillance Systems, AVBS01. Sept 2001