

# EDUM 최종보고서

## (Emergency Detection CCTV Using Machine Learning)

### 문서 정보

구분	소속	성명	날짜	서명
작성자	한국외국어대학교	전진우	2018. 12. 03	
	한국외국어대학교	이대홍	2018. 12. 03	
	한국외국어대학교	임광효	2018. 12. 03	
	한국외국어대학교	권소연	2018. 12. 03	
	한국외국어대학교	김준영	2018. 12. 03	
검토자				
사용자				
승인자	한국외국어대학교	홍진표		

## 머리말

본 문서는 카메라의 영상에 대한 객체 인식을 통해 대단위 아파트, 상가단지 등과 같은 거주지역에서의 이상 상황 감지 및 알림을 전송하는 EDUM 시스템에 대한 상세 설계를 기술한 것이다.



## 개정 이력

버전	작성자	개정일자	개정 내역	승인자
1.0	이대홍 전진우 김준영 임광효 권소연	2018. 12. 03	초안 작성	
	<b>검토자</b>	전진우		
1.1	이대홍 전진우 김준영 임광효 권소연	2018. 12. 05	초안 수정	
	<b>검토자</b>	전진우		

## 목차

1. 개요.....	8
1.1 목적.....	8
1.2 관련문서.....	9
1.3 용어 및 약어.....	9
2. 제품소개.....	10
2.1 제품개요.....	10
2.2 제공 서비스.....	10
3. 시스템 구성도.....	11
3.1 전체 시스템 구성.....	11
3.2 세부 시스템 구성.....	11
3.2.1 수집·제어부.....	11
3.2.2 객체·행동인식부.....	11
3.2.3 Server & DataBase.....	12
3.3 소프트웨어.....	13
3.4 하드웨어.....	14
4. 시스템 상세설계.....	15
4.1 Filter.....	15
4.1.1 VideoCamera.....	15
4.1.2 Frame sender.....	16
4.1.3 Frame scheduler.....	17
4.2 Detection.....	17
4.2.1 Frame Receive.....	18
4.2.2 Object detection.....	18
4.2.3 Act Recognition.....	20

5. 제품기능 설명.....	25
5.1 인터페이스 및 기능설명.....	25
5.2 Class Diagram.....	26
5.3 사용자 흐름도.....	27
5.4 시나리오.....	27
6. 실제적용방안 및 기대효과.....	28
7. 프로젝트 세부추진계획 및 세부일정.....	30
8. Source Code.....	30

## 그림 목차

[Figure1] 시간에 따른 관제능력.....	7
[Figure2] 시스템 구성도.....	10
[Figure3] C922 Pro Stream Webcam / SPC-A1200MB.....	13
[Figure4] Filter Module 구성도.....	14
[Figure5] StoreVideo Code.....	14
[Figure6] 수신 측과의 연결 생성.....	15
[Figure7] 각 카메라의 프레임 전송.....	15
[Figure8] Schedule.....	15
[Figure9] Detection Module 구성도.....	16
[Figure10] 2대의 카메라를 parsing하여 각 객체에 정보 저장.....	18
[Figure11] CNN 연산 과정.....	18
[Figure12] Faster R-CNN 객체 추출 방식.....	17
[Figure13] 객체 인식 결과.....	18
[Figure14] 객체 인식 결과 저장.....	18
[Figure15] 이미지 가공 처리.....	20
[Figure16] 월담 예시.....	21
[Figure17] 월담감지 Code.....	21
[Figure18] Server 구성도.....	22
[Figure19] 상황 별 가중치.....	23
[Figure20] post save signal.....	23
[Figure21] 관리자 경고화면 팝업창.....	24
[Figure22] SMS 문자전송.....	25
[Figure23] Class Diagram.....	27
[Figure24] Sequence Diagram.....	27

[Figure25] 사용자 흐름도.....	28
[Figure26] 세부일정.....	31

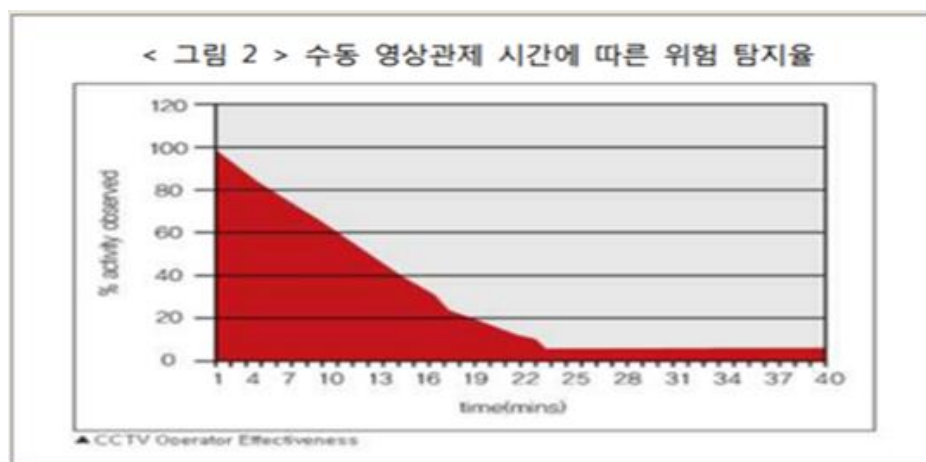
## 표 목차

[Table1] 관련문서.....	9
[Table2] 용어 및 약어.....	9
[Table3] 인터페이스 및 기능 설명.....	26

## 1. 개요

본 장에서는 Inception v2 모델을 이용한 객체 인식을 통하여 아파트 단지 내의 이상 상황들을 감지, 신속한 대처가 가능하도록 알림을 발신하는 시스템인 EDUM의 시스템 및 구성, 기능에 대한 총괄개요를 제공한다. 여기서는 'EDUM'의 목적과 이용 범위, 정의 사항, 참고자료 그리고 본 상세설계서의 개요를 소개한다.

### 1.1 목적



[Figure1] 시간에 따른 관제능력

기존의 보안 시스템은 관리자가 직접 영상을 보며 이상 상황을 판단하는 방식이다. 하지만 [그림]에 따르면, 지속적으로 사람이 직접 영상을 감시할 때, 시간이 지남에 따라 급격하게 관제 능력이 떨어짐을 알 수 있다. 따라서 본 프로젝트에서는 객체 인식을 통해 이상 상황을 판단하여 기존의 방식보다 더 효율적이고 기존의 방식보다 더 적은 비용으로 보안 시스템을 구축하는데 목적이 있다.



## 1.2 관련문서

문서	문서 제목
연구성과 실용화 진흥원	영상 감시 시스템 시장 및 기술동향
한국지역정보개발원	지능형 CCTV 기술 현황 및 활용 사례
한국디지털CCTV연구조합	차세대 지능형 CCTV 산업 경쟁력 강화 방안 연구
전자부품 연구원	지능형 CCTV 시스템 기술 이슈 및 산업동향
Shaoqing Ren, Kaiming He, Ross Girshick, Jian Sun	Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks
Ross Girshick Jeff Donahue Trevor Darrell Jitendra Malik UC Berkeley	Rich feature hierarchies for accurate object detection and semantic segmentation

[Table1] 관련문서

## 1.3 용어 및 약어

용어 및 약어	풀이
COCO dataset	Common object in Context dataset
SMS	Short Message Service
HTML	Hyper Text Markup Language

[Table2] 용어 및 약어

- Faster RCNN: Fast R-CNN의 Region Proposal 방법인 Selective Search 방식을 개선한 방식으로 CNN을 통해 추출된 특징 맵을 RPN에 입력한다. RPN(Region Proposal Networks)에 입력 시 Object가 있을 만한 구역에 대한 Proposal을 연산한다.
- 가상 펜스: 접근 제한 구역에 가상으로 그어진 선으로 이 선을 넘어가는 사람을 감지하면 관리자에게 알림을 보낸다.

## 2. 제품소개

### 2.1 제품개요

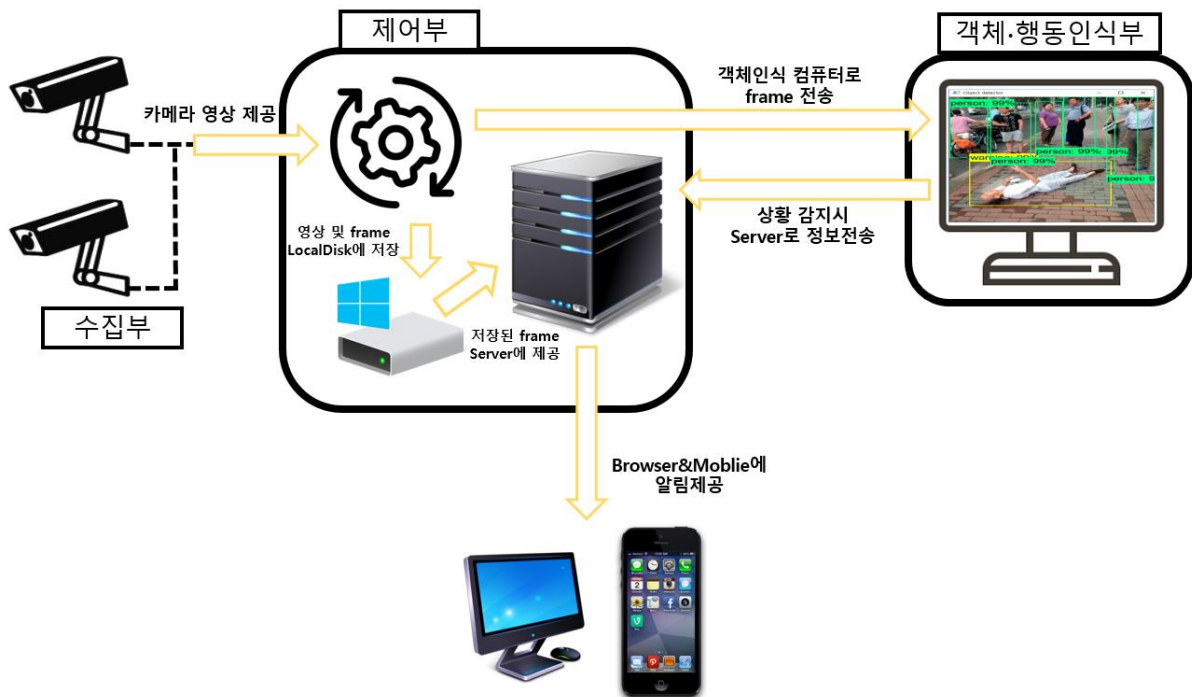
EDUM은 관리 구역 내에서 발생하는 여러 이상 상황을 객체 인식과 상황 감지 시스템을 통해 알림을 제공하여 다수 카메라에 대한 편리한 관리 및 감독을 제공하는 시스템이다. 관리자는 이 알림으로 해당 구역의 실시간 영상을 확인하여 신속하게 현장에 출동할 수 있다.

### 2.2 제공서비스

관리자는 EDUM을 통해 여러 카메라를 한 화면에서 관제할 수 있으며, 그 영상들의 객체 인식 및 상황 감지 중 이상 상황이 감지 되었을 경우 그에 대한 알림을 웹 페이지의 팝업창과 SMS 알림을 통해 제공받을 수 있다. 이 알림에는 감지된 상황의 종류 및 카메라에 대한 정보와 함께 해당 구역의 실시간 스트리밍 영상이 포함되어 있다. EDUM은 주민에게도 제한된 구역에 한해 카메라 영상의 스트리밍 서비스를 제공하여 주민은 근처 장소의 카메라 영상을 볼 수 있다.

### 3. 시스템 구성도

#### 3.1 전체 시스템 구성



[Figure2] 시스템 구성도

#### 3.2 세부 시스템 구성

##### 3.2.1 수집·제어부(Camera Module, Filter)

수집·제어부는 사전에 고정한 FPS로 영상을 제공하는 각 구역에 설치된 카메라를 통하여 각 이미지 프레임을 객체·행동인식부로 전송하는 기능을 수행한다.

Webcam을 통해 받은 영상은 Python, OpenCV 모듈을 사용하여 수집 및 처리하며, 각 webcam별로 수집된 영상의 프레임은 객체·행동인식 결과에 따라 동적으로 바뀌는 스케줄에 따라 순차적으로 객체·행동인식부로 전송한다.

##### 3.2.2 객체·행동인식부(Detection Module)

객체·행동인식부는 수신한 프레임을 통하여 해당 구역의 객체를 인식하고 객체의 상황을 파악하는 기능을 수행한다. Python, Tensorflow 모듈을 사용하여 프레임을 처리한다. 먼저 수집·제어부로부터 수신 받은 이미지 프레임을 토대로 객체 인식

을(사람, 쓰러진 사람, 쓰레기) 하여 결과를 이미지 프레임에 씌워 가공하며, 이후 가공한 이미지 프레임을 가지고 행동 인식(사람이 쓰러짐, 쓰레기 투기, 월담, 접근 제한구역 침입)을 한 후, 결과를 Server로 전송한다. 객체를 인식하기 위해서는 객체 인식 모델이 필요하며, 사전 훈련된 모델이 제공하는 객체 이외의 객체를 인식하기 위해 이미지 샘플을 추가하여 머신 러닝을 수행한다. 추가 학습된 Custom Model을 기반으로 객체 인식을 수행하며, 객체 인식후에 각 객체의 상태와 좌표 값을 이전 프레임과 비교하여 사람이 쓰러졌는지, 쓰레기가 버려졌는지, 사람이 담을 넘는지, 제한 구역에 침입했는지를 확인한다. 그 후 확인된 결과를 토대로 데이터를 Server로 전송하게 된다.

### 3.2.3 Server & DataBase

Server는 객체·행동인식부로부터 전송받은 객체·행동 인식 결과 정보를 DB에 저장하며, 받은 정보를 바탕으로, 연결된 카메라마다의 프레임 가중치를 계산하여 DB에 저장한다. 이 계산된 가중치는 수집·제어부가 보낼 카메라별 프레임 순서를 배치하는 스케줄을 구성할 때 사용된다. 또한, Server는 객체·행동인식 결과로부터 관리자가 확인해야 할 상황을 판단하여 DB에 카메라의 상태를 warning 상태로 업데이트한다. 그리고 warning 상태인 카메라가 감지될 시 server는 관리자의 브라우저에 팝업창을 띄워 관리자에게 감지된 상황 및 감지된 곳의 위치를 알려주며, 감지된 장소 카메라의 영상을 스트리밍을 통하여 보여준다. 또한, 관리자의 부재를 대비하여, 현재 근무중인 관리자의 휴대폰에 SMS를 전송하여, 감지된 상황, 장소의 정보를 제공한다.

### 3.3 소프트웨어

- **Tensorflow**

Tensorflow는 구글이 개발하여 오픈소스로 공개한 기계학습 라이브러리이다. Window나 Linux등 다양한 OS에서 사용이 가능하다. 본 프로젝트는 위 라이브러리 중 Object Detection 파트를 중점적으로 사용하며, 카메라로 촬영한 영상을 처리하여 특정 객체를 인식하는 역할을 한다.

- **OpenCV**

OpenCV는 Open Computer Vision의 약자로 오픈 소스 컴퓨터 비전 라이브러리이다. Window, Linux 등 다양한 OS에서 사용이 가능하다. 실시간 이미지 처리에 중점을 둔 라이브러리로 객체 인식 결과를 바탕으로 즉각적으로 행동을 인식해야 하므로, 본 프로젝트에서 적합하다. 월담 감지를 위한 가상펜스와 객체 추적을 위한 트래커 상자를 그리는데 쓰인다.

- **DataBase**

데이터베이스 관리 시스템으로 서버가 아니라 응용 프로그램에 넣어 사용하는 비교적 가벼운 데이터베이스이다. 대규모 작업에는 적합하지 않지만, 중소 규모라면 속도에 손색이 없고, API는 단순히 라이브러리를 호출하는 것만 존재하며, 데이터를 저장하는데 하나의 파일만을 사용하는 것이 특징이다. 본 프로젝트에서는 이상상황에 대한 정보 및 관리자와 주민에 대한 정보를 저장하는 역할을 한다.

- **Framework**

프레임 워크는 소프트웨어의 구체적인 부분에 해당하는 설계와 구현을 재사용이 가능한 형태로 클래스들을 제공하는 것으로 쉽게 말하면 프로그래밍에서 응용 프로그램 표준 구조를 구현하는 클래스와 라이브러리의 모음이다. 그 중에서 Django는 Python 으로 만들어진 무료 오픈 소스 웹 어플리케이션 프레임워크이다. 특징으로는 구현에 빠르고 웹 개발과 관련된 기능이 많으며 보안에 뛰어나고 확장성이 좋아 다양한 분야에 사용된다. 본 프로젝트에서는 Django를 이용하여 웹서버를 제작하고 Daphne를 사용하여 웹서버를 배포한다.

- **CoolSMS**

CoolSMS는 일반 사용자가 쉽고 빠르게 문자를 발송할 수 있도록 지원하는 서

비스로 API를 통하여 SMS를 사용자에게 전송한다. 경고 알림과 동시에 장소, 이상상황에 대한 정보를 로그인 되어 있는 관리자에게 전송하는데 사용된다.

### 3.4 하드웨어

- Webcam

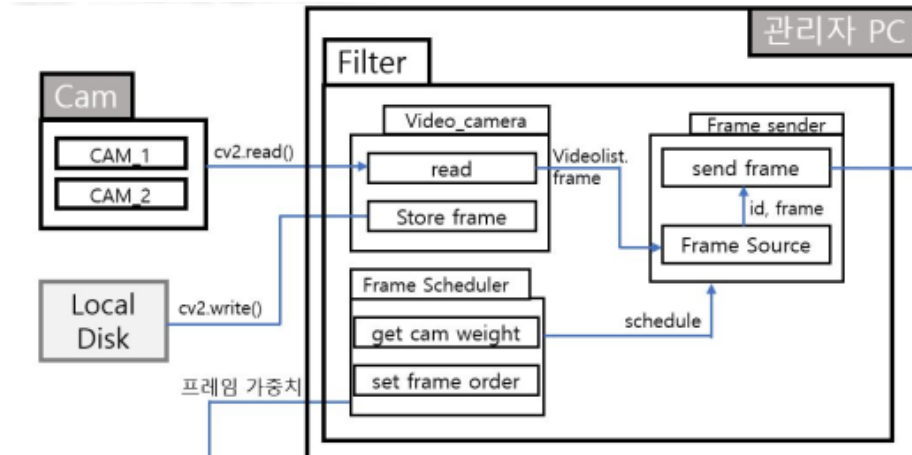
아파트 단지 내에 설치하여 해당 구역의 상황을 촬영하는 카메라로, 영상을 끊김 없이 스트리밍해야 하기 때문에, 고성능 카메라인 C922 Pro Stream Webcam을 사용한다. 보조 카메라로 삼성 SPC-A1200MB를 사용한다.



[Figure3] C922 Pro Stream Webcam / SPC-A1200MB

## 4. 시스템 상세설계

### 4.1 Filter



[Figure4] Filter Module 구성도

#### 4.1.1 VideoCamera

컴퓨터에 연결된 camera를 실행시켜준다. 현재시간을 기준으로 output영상 객체를 카메라마다 각각 생성해주고 생성된 객체에 지속적으로 frame을 써 주면 동영상으로 저장된다. 저장을 시작한 시간을 기준으로 사용자가 설정한 시간이 지나면 동영상 저장 객체를 중지시켜 저장을 완료한다. 그와 동시에 동영상 저장이 끊이지 않도록 즉시 output영상 객체를 재생성 후 새로 저장을 시작한다. 또한 모든 동영상 저장은 각 camera에 할당된 id를 통해 각각 저장된다.

```
self.video_list[cam_id].sizecon()
if self.video_list[cam_id].sizecontrol % 4 == 0: # 영상 용량 조절
    self.video_list[cam_id].storeframe(self.video_list[cam_id].frame)
if (time.time() - self.video_list[cam_id].time) > 5: #영상 저장 시간 조절
    self.video_list[cam_id].__del__() # 현재까지 영상 저장
    self.video_list[cam_id].write(cam_id) # 영상저장함수실행
```

[Figure5] StroeVideo Code

위 코드에서 볼 수 있듯이 sizecontrol변수를 이용하여 초당 저장하는 프레임 수를 조절하며 (현재시간 - 동영상의 저장시간)으로 동영상 저장시간을 조절할 수 있다.

압축은 동영상에 저장하는 frame을 [cv2.IMWRITE\_PNG\_COMPRESSION, x]를 이용하여 png파일의 압축률을 조절할 수 있었지만 동영상의 용량차이는 크

지 않았으며, 오히려 용량은 비슷하지만 영상을 읽어오는 속도와 저장속도가 눈에 띄게 느려져 선택하지 않았다. 실제로는 초당 frame저장비율이 영상의 용량에 큰 차이를 보임을 알 수 있었다. 따라서 저장 시 sizecontrol변수를 통해 초당 저장하는 frame수를 조절하여 영상의 용량을 조절하였다.

#### 4.1.2 Frame sender

**VideoCamera**를 통해서 얻은 프레임을 객체.행동인식부로 전송하는 클래스이다.

먼저 initialize\_server()를 통해 수신 측인 Frame\_receive와 tcp 소켓연결을 생성하고, Frame을 보내기전 연결되어 있는 카메라들의 정보를 보내고, Frame\_receive 측이 정보를 잘 받았다는 답신이 오면, 카메라의 프레임 전송을 시작한다. 연결된 카메라의 수만큼 VideoCamera객체를 생성하여 각 카메라마다 프레임을 한 장씩 받아오며, 총 읽어오는 프레임은 각 카메라마다 초당 10프레임이다. 각 카메라로부터 읽어온 프레임은 frame\_list에 저장되며, frame\_list에서의 인덱스는 카메라의 id와 같다. Frame sender는 Frame scheduler객체를 내부에서 생성하고, 주기적으로 Frame scheduler객체가 생성한 schedule의 순서에 맞게 frame\_list안의 카메라의 프레임과 그 프레임이 어느 카메라로부터 온 것인지 알려주는 카메라 id정보를 send\_frame()을 통해 전송한다. 이 때 한 번의 전송은 한 카메라에 대한 프레임과 id정보이다.

```
b'OK'
Send Complete
Send Complete
Connect Success
[2, 1]
```

[Figure6] 수신 측과의 연결 생성

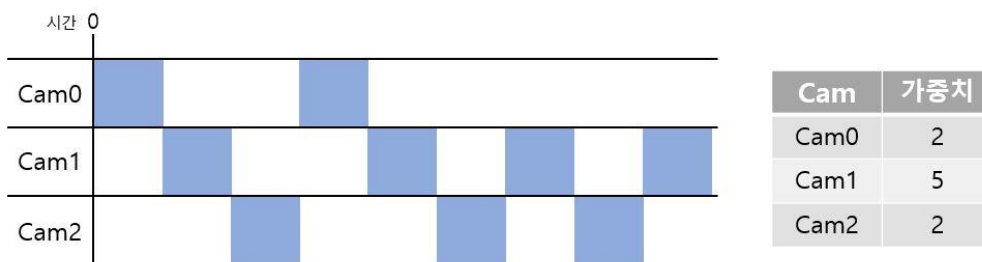
```
##SIZE0##: 16706
Echo cam num: b'0'
Send comp0
##SIZE1##: 12223
[0, 1, 0]
```

[Figure7] 각 카메라의 프레임 전송



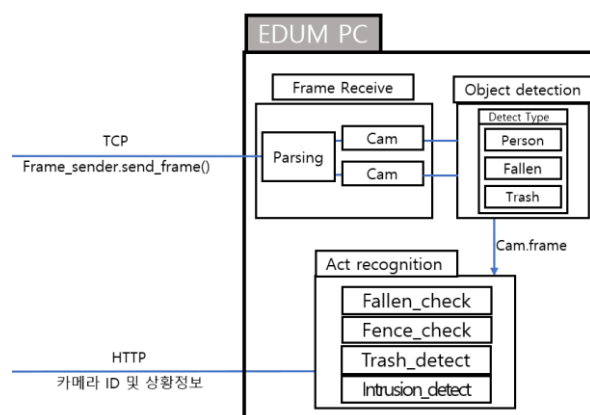
### 4.1.3 Frame scheduler

**Frame scheduler**는 server가 객체·행동인식 결과로부터 각 카메라마다 부여하고 DB에 저장한 가중치 정보를 이용하여, 각 카메라의 프레임들이 전송될 순서를 스케줄링하여 Frame sender로 이 스케줄링 정보인 schedule를 제공하여, 우선 순위가 높은 상황을 더 많이 **객체·행동인식부**로 전송하게 한다. 스케줄링 방식은 라운드 로빈 방식을 채택하여, 한 카메라는 한 번에 한 프레임만 보낼 수 있도록 제한한다. 그리고, 각 카메라의 가중치에 비례하여 서비스 시간을 다르게 주어, 결과적으로 가중치가 높은 카메라일수록 단위 시간 동안 더 많은 프레임을 **객체·행동인식부**로 전송한다. 이러한 방식을 채택한 이유는 행동인식은 객체 인식 결과 정보를 바탕으로 수행되므로, 인식되는 프레임의 순서가 아닌, 같은 시간동안 얼마나 더 많이 객체 인식되어, 얼마나 더 많은 정보를 얻는가에 따라서 행동인식의 정확도가 달라진다고 판단하여, 이와 같은 방법을 채택하였다.



[Figure8] Schedule

## 4.2 Detection



[Figure9] Detection Module 구성도

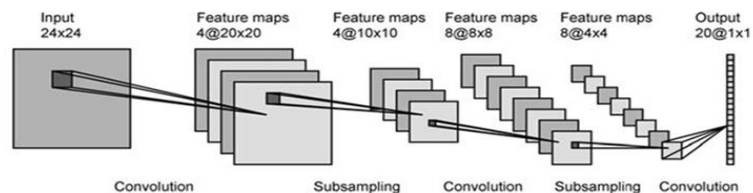
#### 4.2.1 Frame Receive

Filter Module과 TCP 통신을 사용하여 데이터를 송수신하는데, Frame을 수신하기 전, Filter Module으로부터 연결된 카메라의 총 대수만큼 카메라 객체를 생성하고, 생성한 각 객체의 id 및 data를 조작하기 위해 리스트에 shallow copy하여 저장한다. 이후 객체의 id로 수신한 프레임을 parsing하며, 객체의 frame 변수에 프레임을 저장 후 Object detection Module로 전달한다. 아래와 같이 각 객체에 정보가 저장되어진다.

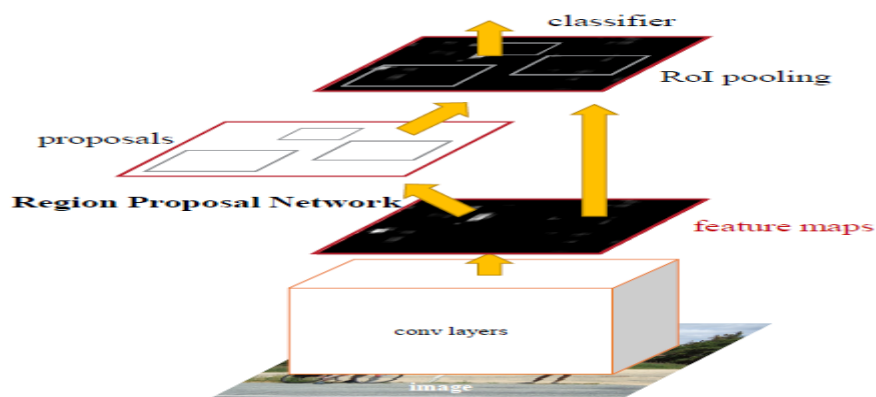
```
Socket bind complete
Socket now listening
Number of CCTV: 2
Parsing the CAM INFO...
Cam_id: 0
restricted: False
COMPLETE
Parsing the CAM INFO...
Cam_id: 1
restricted: False
```

[Figure10] 2대의 카메라를 parsing하여 각 객체에 정보 저장

#### 4.2.2 Object detection



[Figure11] CNN 연산 과정



[Figure12] Faster R-CNN 객체 추출 방식

객체 인식에 사용된 Inception V2 모델은 Faster R-CNN 알고리즘을 사용하여 객체를 인식한다. 비교 모델인 YOLO V3 모델이 더 빠른 연산을 제공하지만 Inception V2 모델이 더 세밀한 객체의 인식률이 높다고 판단하여 사용하였다.

```

person
dict_keys([1, 2, 3, 4, 5])
fallen
dict_keys([1, 2, 3, 4, 5])
trash
dict_keys([1, 2, 3, 4, 5])
trash
dict_keys([1, 2, 3, 4, 5])

```

**[Figure13] 객체 인식 결과**

```

(boxes, scores, classes, num) = self.sess.run(
    [self.detection_boxes, self.detection_scores, self.detection_classes, self.num_detections],
    feed_dict={self.image_tensor: self.frame_expanded})

```

**[Figure14] 객체 인식 결과 저장**

Frame Receive Module로부터 전달받은 프레임으로 객체인식을 수행한다. 객체는 person, fallen, trash, bottle, metal 5개가 있으며, 인식한 객체의 경계선, 정확도, class name과 인식한 객체의 개수가 변수에 저장된다. 이후 저장된 변수를 토대로 프레임 이미지에 씩워 가공하며, 가공 처리된 프레임을 Act Recognition Module로 전달한다.



**[Figure15] 이미지 가공 처리**

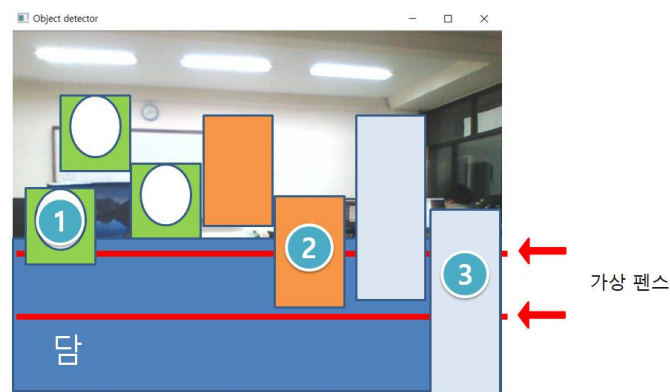
### 4.2.3 Act Recognition

월담, 접근 제한 구역 침입, 쓰레기 투기의 세 가지 상황을 감지하는 모듈로, OpenCV의 다수 객체 추적기(MultiTracker)와 객체인식 cam객체의 객체 좌표를 이용하여 상황 발생 여부를 감지한다. 월담을 감지하는 fence\_detect, 접근 제한 구역 침입을 감지하는 intr\_check, 쓰레기 투기를 감지하는 Trash\_detect의 세 함수를 통해 상황을 감지한다.

- .fence\_detect함수

월담을 감지하는 함수로 담이 있는 구역에 한해 실행된다. 사람 객체를 추적할 객체 추적기 fence\_tracker 변수를 설정할 fence\_settings 함수와 새 프레임마다 추적기를 업데이트하고 월담을 감지했는지 좌표와 경계선 값을 비교하는 fence\_updates함수를 사용한다. 각 함수에서는 모두 object detection module의 카메라객체를 인수로 받아 카메라의 프레임, 사람 경계 박스 좌표 리스트를 사용한다. 객체 추적기 fence\_tracker는 사람 객체가 인식됐고 사람 수가 변경되었을 때 초기화되며 사람 수의 변경은 없지만 사람이 계속 인식될 때에는 fence\_updates를 통해 월담 여부가 계산된다. 이외의 상황에서는 사람 객체가 인식되지 않으므로 객체 추적기는 비활성화된다.

월담을 감지하는 알고리즘은 세 가지 경우에 나눠 실행한다. 담 뒤의 사람이 인식될 경우, 담 앞의 사람이 상체만 인식될 경우, 담 앞의 사람의 전신이 인식될 경우로, 두 개의 가상펜스 선을 그려 월담 여부를 각 객체의 좌표 위치와 비교한다.



[Figure16] 월담 예시

위 그림에서 각 상자는 사람 객체 경계박스를 따라 추적하고 있는 추적 상자이다. 담의 윗부분과 가운데 부분에 가상 펜스를 설치하고, 각 객체 상자의 아래쪽 y 좌표 위치와 위쪽 y좌표 위치를 비교하여 그 위치가 이전 프레임과 비교하여 변화가 있을 경우 월담으로 감지한다.

```
for index, i in enumerate(temp):
    if index < len(self.fence_prev):
        # 담 뒤에 있던 사람이 앞으로 간 것으로 위치가 바뀐 경우 월담 감지
        if self.fence_prev[index][4] == 0 and y < self.colist1[1]:
            self.fence_warning = True
            print("뒤 -> 앞 월담 감지")
            i[4] = 2
        # 담 앞에 있던 사람이 뒤로 간 것으로 위치가 바뀐 경우 월담 감지
        elif self.fence_prev[index][4] == 1 and y > self.colist2[1]:
            self.fence_warning = True
            print("앞 -> 뒤 월담 감지 (몸 전체 인식)")
            i[4] = 0
        # 담 앞에 있던 사람이 뒤로 간 것으로 위치가 바뀐 경우 월담 감지
        elif self.fence_prev[index][4] == 2 and y > self.colist1[1]:
            self.fence_warning = True
            print("앞 -> 뒤 월담 감지 (상체부분 인식)")
            i[4] = 0
```

[Figure17] 월담감지 Code

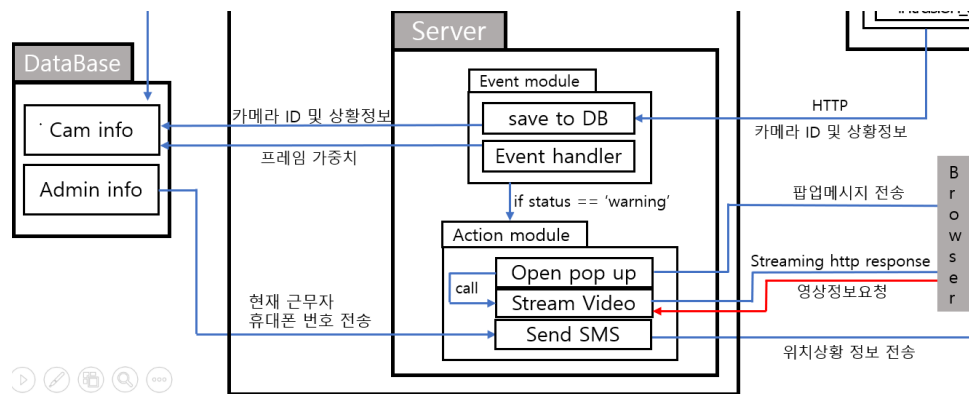
- intr\_check함수

접근 제한 구역 침입의 경우 관계자 외의 사람 혹은 사람 출입이 완전히 제한되는 구역을 감시하고 있도록 설치된 카메라의 화면에서 사람 객체가 인식될 경우 제한 구역 침입으로 감지한다.

- Trash\_detect함수

쓰레기 투기의 경우 쓰레기와 사람이 모두 인식될 경우, 쓰레기만 인식될 경우로 나누어 쓰레기 무단 투기를 감지한다. 쓰레기와 사람이 모두 인식될 때, 사람과 쓰레기의 객체 인식 경계 박스가 겹치면 그 사람을 쓰레기를 들고 있는 사람으로 판단, 경계 박스의 인덱스를 저장한다. 이후 그 두 객체를 추적하여 추적 상자의 중점 간 위치가 멀어지면 쓰레기 투기로 감지한다. 이 외에도 쓰레기만 단독으로 인식될 경우 10초 이상 연속으로 인식되면 쓰레기 투기로 감지하며, 두 상황을 제외한 상황에서는 객체 추적기가 비활성화된다.

## 4.3 Server



[Figure18] Server 구성도

Server는 객체·행동인식부로부터 받은 정보를 저장하고, 관리자가 확인해야 할 상황을 판단하여 DB에 카메라의 상태를 warning 상태로 업데이트하는 Event module과 카메라의 상태가 warning인 이벤트 발생시 팝업창을 띄워 관리자에게 감지된 상황 및 감지된 곳의 위치를 알려주며, 감지된 장소 카메라의 영상을 스트리밍을 통하여 보여주고, 또한, 현재 근무중인 관리자의 휴대폰에 SMS를 전송하여, 감지된 상황, 장소의 정보를 제공하는 Action module로 구성 되어있다.

#### 4.3.1 Event module

객체·행동인식부로부터 Post request를 통해 받은 인식 결과 값을 저장하는 부분이다. Django에서는 DB 테이블의 접근을 modelname.objects.all()을 통해서 할 수 있으며, 리턴 된 객체를 통해 DB테이블의 attribute의 값을 읽어오거나, 수정할 수 있다. DB로의 저장완료후, event\_handler 함수를 호출하여, 저장된 인식 결과 값을 통해, 각 카메라마다 가중치를 부여한다.

```
cam.weight = 1 # default priority of camera
if cam.fallen:
    cam.weight += 2
if cam.trash:
    cam.weight += 1
if cam.instrusion:
    cam.weight += 3
if cam.fence:
    cam.weight += 3
```

[Figure19] 상황 별 가중치

가중치의 기본 값은 감지상황이 아무 것도 없을 경우에는 1이며, 쓰러진 사람이 감지될 시에는 2만큼 증가하고, 쓰레기 감지때는 1, 접근 금지 구역 침입 감지 및 월담 행위는 3만큼 증가한다. 각 카메라마다 위의 4가지 상황 중 하나라도 감지가 된다면, 그 카메라는 warning 상태가 되며, action module이 이를 감지하

여, 관리자에게 알림을 줄 수 있는 action들을 수행하게 된다.

#### 4.3.2 Action module

Django에서는 어떤 특정한 일이 수행되면, 이를 알려주는 signal을 발생하는 기능을 가지고 있다. 그 중 post\_save signal를 이용하여 save()를 통해 DB값이 변화되었을 때를 감지하여, 카메라의 status가 warning일 경우 관리자의 브라우저에 감지상황과 감지된 카메라의 위치를 websocket을 통해 전송하여, 그 내용이 표시되도록 하며 추가로 팝업창을 띄워, 상황이 감지된 카메라의 영상을 제공한다.

```
@receiver(post_save, sender=Camera)
def send_detect_info(sender, instance, created, **kwargs):
```

[Figure20] post save signal

- **Websocket**

관리자가 직접 브라우저를 새로고침하지 않더라도 서버로부터 받은 정보를 실시간으로 볼 수 있게 하기위해 websocket을 사용하였다. Django에서는 websocket을 channels 모듈로 지원을 한다.

channel\_layer.group\_send()를 통해 감지상황과 상황이 감지된 카메라의 정보들을 관리자의 웹 브라우저로 보내며, 관리자의 웹 브라우저는 javascript를 통해서 실행되고 있는 websocket을 통해 정보를 전송받아, 받은 정보를 브라우저에 표시하고, 팝업창을 띄워, streaming을 통해 감지된 카메라의 영상을 관리자에게 보여준다.

- **Streaming**

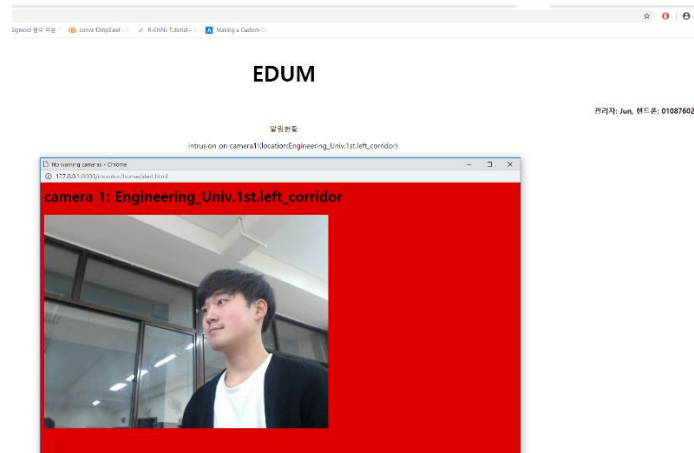
- ✓ **frame저장**

관리자와 주민에게 영상을 실시간으로 Streaming하기 위해 camera와 연결된 컴퓨터에서는 camera의 frame을 LocalDisk에 지속적으로 저장한다. 저장시 5개의 frame을 순서대로 저장하며 5개가 전부 저장되고 나면 제일 이전에 저장된 frame부터 덮어쓰며 저장한다.

- ✓ **Streaming**

alarm을 울릴 때 서버에서는 StreamingHttpResponse()를 통해 LocalDisk에

저장되어있는 frame을 지속적으로 읽어와 Browser에 보여준다.



[Figure21] 관리자 경고화면 팝업창

#### ✓ 싱크조절

LocalDisk에 저장되는 frame은 5개이며 지속적으로 덮어쓰워진다. 1초에 약 30frame을 저장하므로 5개의 frame은 약 0.1~2초사이에 저장된다. 이 frame들을 서버에서 읽어올 때 저장된 순서대로 읽어오며 5개의 frame을 전부 다 읽었다면 다시 처음으로 돌아가 순서대로 frame을 읽어온다. 5개의 frame들은 앞서 말했듯이 시간차이가 0.1~2초도 나지 않으며 frame저장시 압축률과 서버에서 frame을 읽어오는 속도를 조절하여 실제로 streaming되는 영상을 볼 시에는 거의 끊김과 delay가 없는 영상을 확인할 수 있다.

#### ✓ 주민과 관리자

주민과 관리자가 같이 streaming되는 영상을 확인할 수 있어야 한다. 하지만 주민은 원하는 camera의 영상을 선택하여, 관리자는 warning상태의 camera만을 확인해야 하는 차이가 있다. 이는 streaming을 제공하는 함수를 읽어오는 경로를 바꾸어 각각 생성하여 주민페이지와 관리자 alarm페이지에 각각 매칭해줌으로써 해결하였다.



## ● SMS



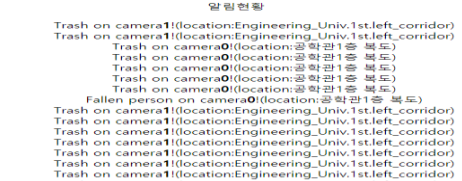
SMS 전송을 제공하는 CoolSMS의 API를 사용한다. 이상상황이 발생하는 경우 이상상황과 장소에 대한 정보를 DB에서 받아 웹서버에 로그인 되어 있는 관리자에게만 SMS를 전송한다.




[Figure22] SMS 문자전송

## 5. 제품기능 설명

### 5.1 인터페이스 및 기능 설명

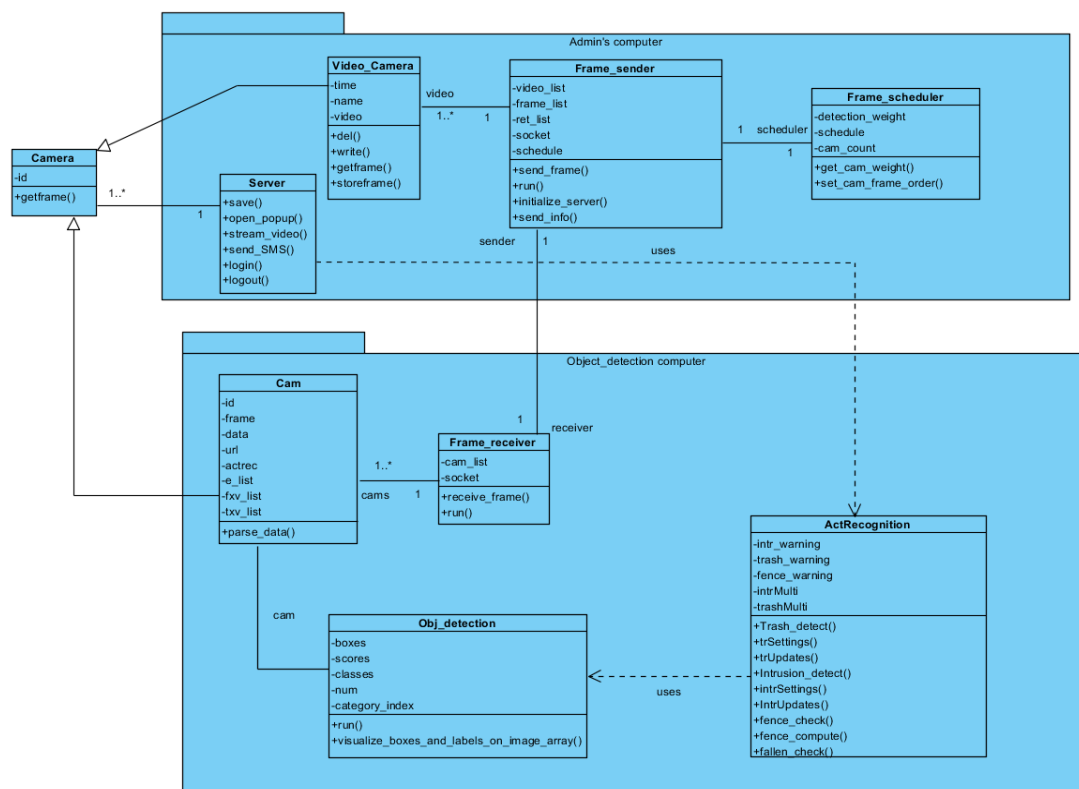
기능	설명	User Interface
로그인	로그인 기능을 통하여 관리자는 관리자 페이지로 접속하여 알림을 제공받을 수 있으며 일반주민은 원하는 영상을 볼 수 있는 페이지로 넘어간다.	
이상상황 알림 및 영상제공	관리자는 이상상황에 대한 알림을 팝업창으로 확인하며 해당 영상을 제공한다.	
이상상황 기록	이상상황에 대한 알림을 기록으로 남긴다.	

문서명: EDUM 최종보고서

제한된 영상제공	웹페이지를 통하여 원하는 스트리밍 서비스를 선택하여 볼 수 있다.	<p>현재 보고계신 화면은 1번카메라입니다.</p> 
SMS 제공	로그인된 관리자에 한하여 이상상황에 대한 정보와 장소에 대해 서비스를 제공받는다.	<p>[Web발신] 공학관1층 복도 카메라에 월담 행위 감지가 발생하였습니다.</p> <p>[Web발신] 공학관1층 복도 카메라에 월담 행위 감지가 발생하였습니다.</p> <p>도전도1층</p> <p>+ 메시지를 입력하세요</p>

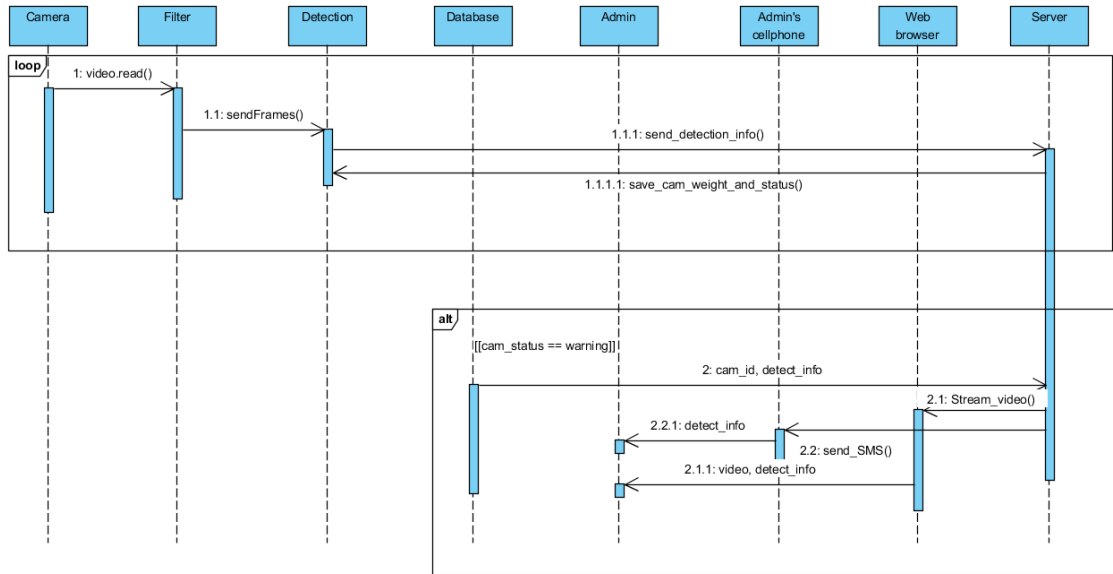
[Table3] 인터페이스 및 기능 설명

## 5.2 Class Diagram



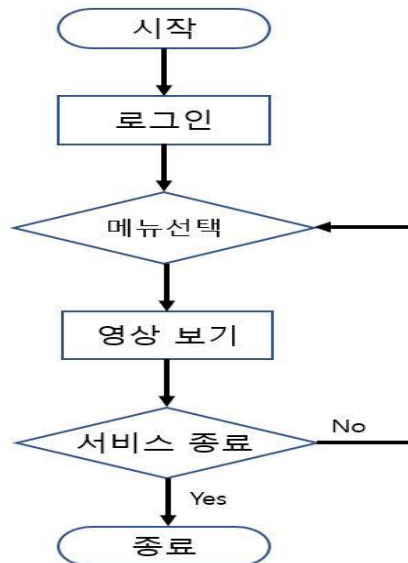
[Figure23] Class Diagram

### 5.3 Sequence Diagram



[Figure24] Sequence Diagram

### 5.4 사용자 흐름도



[Figure25] 사용자 흐름도

## 5.5 시나리오

### Case1: 접근 제한 구역 사고 예방

비상 상황을 대비하여 잠가 놓지 않은 옥상으로 어린 아이들이 계단을 오르고 있다. 계단의 가상 펜스를 넘은 것이 CCTV에 촬영되자 바로 관리자의 웹 페이지에 해당 상황을 알리고 영상을 확대하여 보여준다. 그러한 한 편에서는 순찰 시 관리자의 부재를 대비하여 SMS를 통한 경고 메시지도 전송된다. 관리자는 메시지에 포함된 카메라 위치로 직접 출동하여 옥상으로의 출입을 제한 및 사고를 예방할 수 있다.

### Case2: 무분별한 쓰레기 무단 투기 예방

쓰레기장이 아닌 장소에 20L 봉투에 담긴 쓰레기가 무단으로 버려진다. 10분이 지나도 버린 사람이 회수하지 않고 방치될 경우 해당 장소의 카메라 위치, 영상, 사진을 웹 페이지 및 SMS를 통해 알림을 보낸다. 관리자는 영상을 확인한 뒤 추가적인 쓰레기 무단 투기를 예방하기 위해 바로 해당 장소로 가거나, 순찰 시 그 장소에서 쓰레기를 치우는 등의 대처가 가능하다

### Case3: 사람이 쓰러진 상황의 대처

평소 심장 질환을 앓고 있던 노인이 밤에 외진 곳을 걸어가다 급작스러운 심장 발작으로 인해 자리에서 정신을 잃고 쓰러진다. 10초가 지나도 쓰러진 사람이 일어나지 않자 관리자에게 경고 상황의 알림을 보내고, 확대된 영상을 본 관리자는 상황을 판단하여 구급차를 부른다. 빠른 대처를 통해 골든 타임 안에 환자가 응급실로 이송될 수 있었다

### Case4: 주민의 카메라 확인

아파트의 주민은 아파트에 입주할 당시 아파트 관리 홈페이지의 guest계정을 전달 받는다. 아이들이 놀이터나 공원에서 사고 없이 잘 놀고 있는지 확인하고 싶은 부모는 관리 홈페이지에 들어가 guest계정으로 로그인 후 공원 camera버튼을 클릭 후 공원의 영상을 직접 확인하여 아이들이 안전하게 노는 것을 확인하였다.

## 6. 실제적용방안 및 기대효과

본 시스템은

- 상가단지, 다세대주택 혹은 아파트단지 등에서 사용 가능한 이 시스템은 주민의 편의와 안전을 위해 이용된다.
- 객체 인식을 통한 쓰레짐 감지를 통하여 인적 드문 곳에서의 안전사고 등을 빠르게 대처 가능하다.
- 또한 쓰레기를 감지하여 무단 투기된 쓰레기가 있을 경우, 관리자에게 알려 해당 구역을 빠르게 청소할 수 있다.
- 제한구역을 설정하여 해당지역에 사람이 감지될 경우 즉시 관리자에게 알림을 줘서 통제 가능하도록 한다.
- 펜스나 담을 넘는 등의 행동이 감지될 경우도 제한구역 침입과 마찬가지로 즉시 관리자에게 알림을 줘서 통제한다.
- 위 사항들을 통해 해당 구역 관리자의 관리능력이 저하되지 않고 지속적인 관리가 가능하도록 도와준다.

등의 효율 및 편의성 극대화를 통해 해당 시스템 이용자의 편의를 도모하며 안전 또한 보장이 가능하다. 이를 통해 다른 제품들과 비교해 경쟁력을 가지며 EDUM 시스템의 시장 점유율을 높여 수익창출을 이룬다.

## 7. 프로젝트 세부추진계획 및 세부일정

프로젝트 기간	2018.10.04 ~ 2018.12.15											
개발내용	프로젝트 기간											
	1주	2주	3주	4주	5주	6주	7주	8주	9주	10주	11주	12주
아이디어 회의 및 계획												
프로젝트 관련 자료 및 장비조사												
역할분담 및 핵심기술 조사												
사업제안서 작성												
요구사항 정의서 작성												
서버 및 데이터베이스 구현												
객체 추가학습												
영상 스트리밍 구현												
구역에 따른 상황판단 구현												
Web Service 구현												
개별 테스트 및 보안												
통합 테스트 및 보안												
최종발표 및 시연												

[Figure26] 세부일정

## 8. Source Code

-zip파일 별도 첨부