[Inha University]
12131334 백동원 | 12141466 정선우 | 12163651 허정인

인하대학교 전기종합설계 Project

SVM 알고리즘을 통한 화재 감지 시스템

CONTENTS

01 Existing fire detection system 기존 감지 체계

02 Solving Problem 문제 해결방법

03 Research contents 연구내용

04 Conclusion 결론

05 Test 시연

화재 감지기의 종류

단독형 감지기

독립형 감지기

차동식 감지기

금속 수축/팽창

정온식 감지기

열 감지

광전식 감지기

연기 감지

불꽃 감지기

불꽃 감지

기존 감지 체계의 원리

인체 감지기
Human Rejection

불꽃 감지기
Fire Detection

태양광 감지기
Sunlight Rejection

화재보

기존 감지 체계의 특징

- 오염에 강하고 신뢰성과 경제성이 높음

- 파손에 취약하며 인체의 움직임이나 할로겐, 태양광 등에 오판/오검출할 우려 높음

- 특히 조도, 그림자 등에 가려진 인체의 움직임과, 불이 인체에 일부 가려짐에 따라 오진율이 높아짐.
- 이는 현실 속의 변화하는 상황에 대처하기가 어려움을 뜻함.

문제점

1. 감지기 내 3가지 센서의 경제적 문제

- 하나의 센서라도 고장이 나면 전체를 교체해야 되어 경제적으로 부담
- 불꽃 감지기의 경우 다른 감지기에 비해 비용이 상당함
- 카메라를 이용한 경우, 경제성의 문제와 부피가 큼

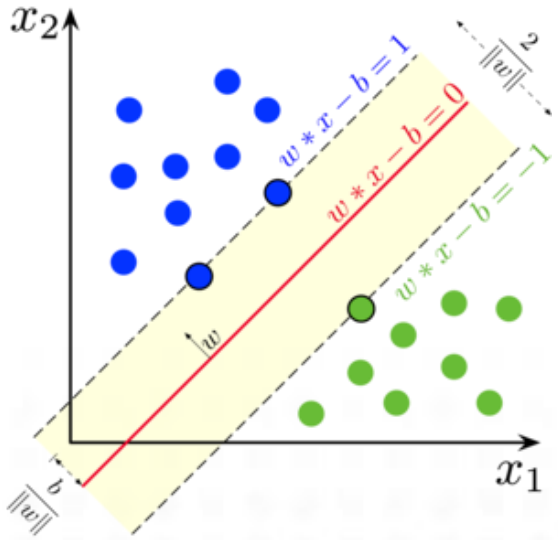
2. 회로의 복잡성 및 오진 문제

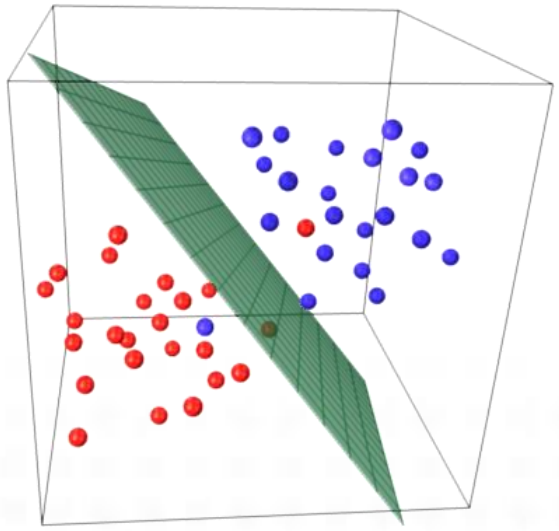
- 회로의 복잡성으로 인한 사후 문제점 발생
- 외부 변화하는 상황으로부터의 오진 발생
- 태양광이나 인체의 움직임을 Flickering(플리커링) 현상으로 인지

3. 시간 복잡도 문제

- FFT 혹은 WAVELET
- 실시간 영상처리에 대해 고성능 PC가 필요

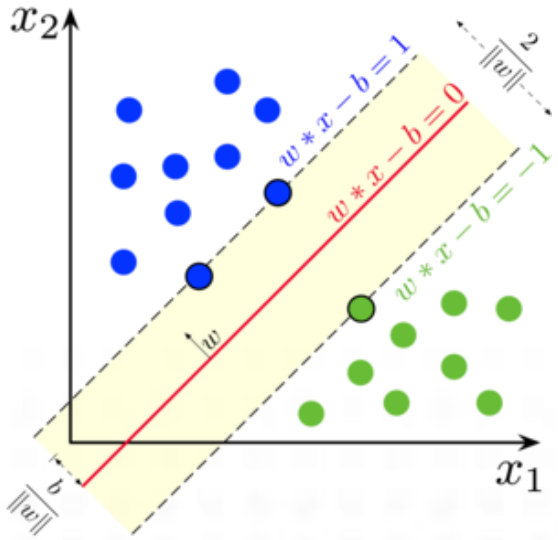
SVM(Support Vector Machine) 알고리즘

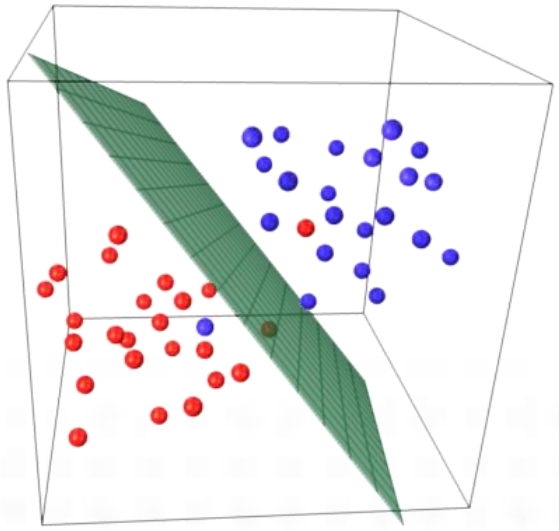




- Machine Learning 중 Supervised Learning (지도학습)의 일종
- 다른 두 데이터 요소 사이에서 마진을 최대화하는 목표를 중점으로 함
- 많은 이미지를 입력할수록 학습량은 높아지고 분류 적중률이 향상

SVM(Support Vector Machine) 알고리즘





- 불꽃(Positive)과 불꽃이 아닌 것(Negative)을 명확히 구분
- 화재 감지 시스템에서 요구하는 이진화 된 검출 조건에 가장 적합
- 불꽃 학습에만 집중하여 저전력 초소형 MPU로도 화재 검출 가능

연구 구성 및 내용

Image processing

Machine Learning

Hardware

영상처리

Image Processing

기존 화재 감지 체계를 대신할 영상처리

- OpenCV
- 라즈베리파이 3B 모델
- 라즈베리파이 공식 카메라 V2.1
- HOG 기술자
- 화재 검출

영상처리 – OpenCV

OpenCV

다중 플랫폼에서 실행할 수 있는 컴퓨터 비전 어플리케이션을 개발하기 위한 오픈소스 라이브러리

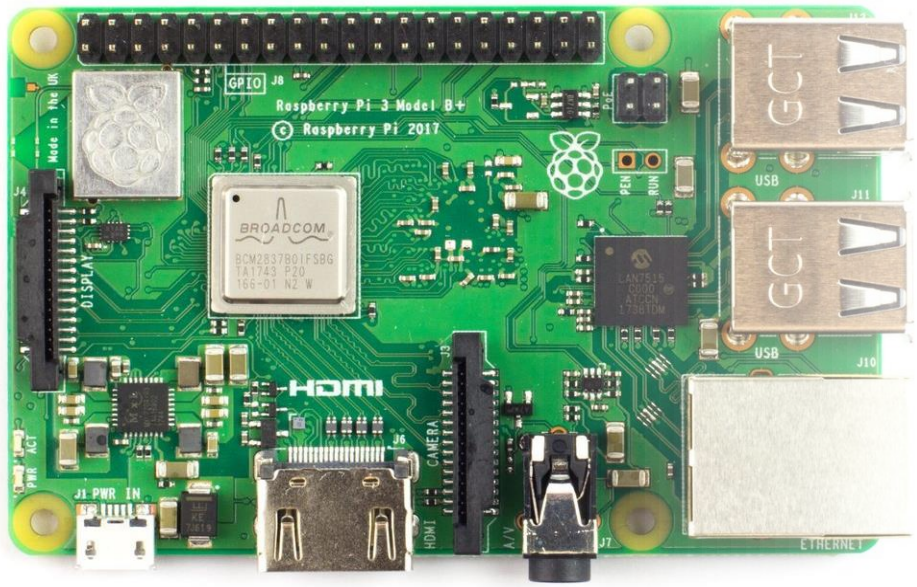
* 다중 플랫폼: 윈도우, 리눅스, MacOS 등

Python

OpenCV는 크게 Python 언어와 C++ 언어를 지원

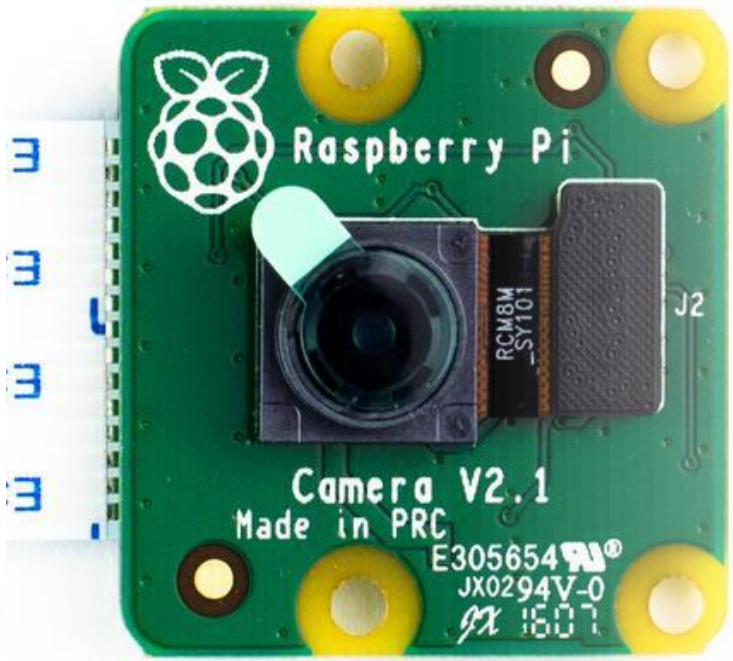
본 연구에서는 이식성이 비교적 뛰어난 Python 언어를 채용

영상처리 – 라즈베리파이 3B 모델



- 중앙처리장치로 라즈베리파이 3B 모델 이용
- CPU 클럭은 1.2GHZ, SRAM은 1GB로 딥러닝 영상처리를 하기에 빈약한 SPECULATION
- 사이즈가 작으며, 가격이 저렴
- 산업 현장에서 일정 정확도를 유지한 채로 부피가 작아진다는 것은 **작업 공간 확보에 더욱 유리함**을 의미

영상처리 – 라즈베리파이 공식 카메라 V2.1



- 센서의 역할로 라즈베리파이 공식 카메라 이용
- 현 연구에서의 라즈베리파이 V2.1 카메라는 8MP의 수동 초점 조절 카메라 이용
- 라즈베리파이와 가장 호환성이 좋으며 동시에 경제적

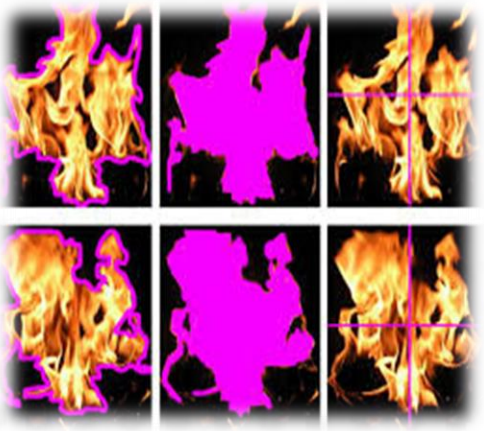
영상처리 – HOG(Histogram of Oriented Gradients) 기술자

- HOG 기술자: 경사지향 히스토그램으로, 유용한 정보를 추출하고 관계가 없는 정보를 버림으로써 이미지를 단순화 또는 이미지 패치를 표현한 것
- 이미지 인식 및 객체 감지와 같은 작업에 유용한 벡터를 이용해 본 연구에 있어 유용한 불의 정보만을 획득 가능

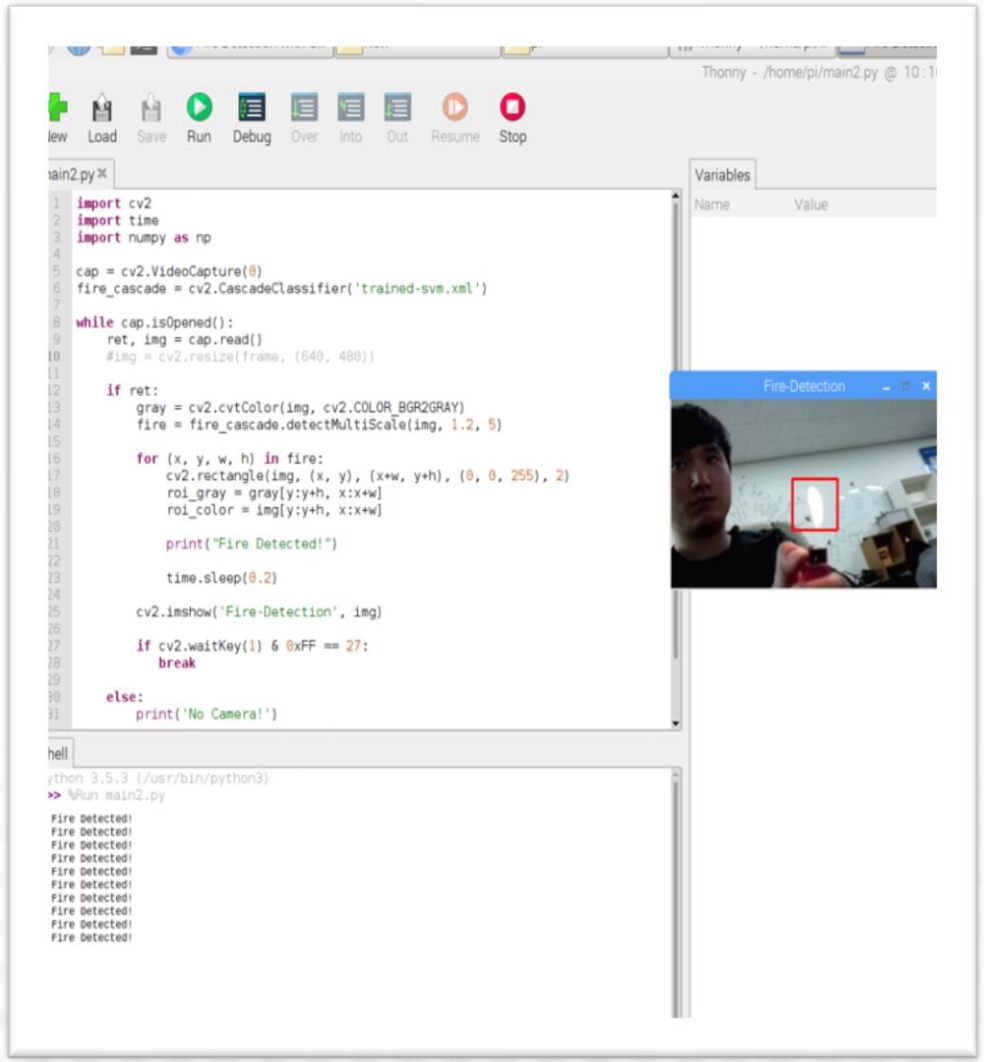
$$g = \sqrt{g^{2}_{x} + g^{2}_{y}}$$

$$\theta = \arctan \frac{g_y}{g_x}$$

- 오른쪽 식을 이용해 그라디언트를 구해낼 수 있음
- 화재는 그 형태가 플리커링으로 인해 지속적으로 바뀌면서도 일정한 특징 조건을 갖추고 있어, HOG 기술자를 적용하기에 적합



영상처리 – 화재 검출



최종 소스코드에서는 일반 라이터를 사용하여 화재 검출을 제대로 실시하는지에 대해 시험

소스 코드가 실행되면 미리 학습된 SVM 알고리즘이 켜지며,
라즈베리파이 카메라를 통해 영상 스트리밍 시작

HOG 기술자와 SVM을 통해 훈련한 가중치와 일치하는 ROI(Region Of Interest)를
사각형 테두리로 둘러싸며 "**FIRE DETECTED**" 메시지 출력

머신러닝

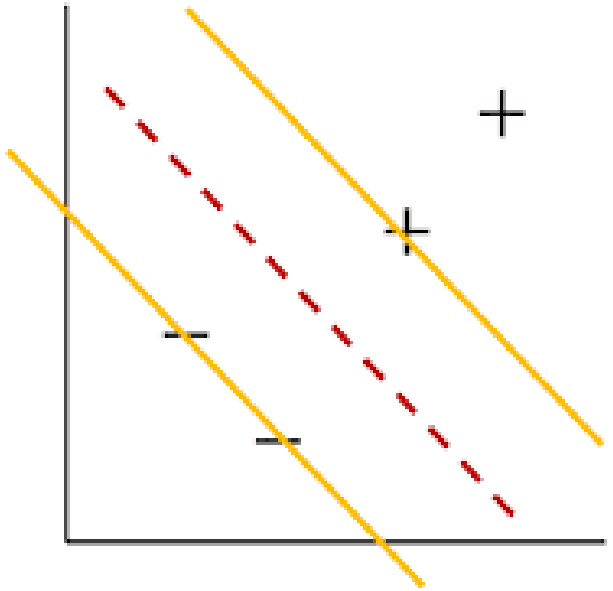
Machine Learning

Machine Learning

변화하는 외부상황에서도 정확하게 감지해낼 수 있도록 하는 SVM(Support Vector Machine)의 설계에 대한 내용

- Support Vector Machine
- Training

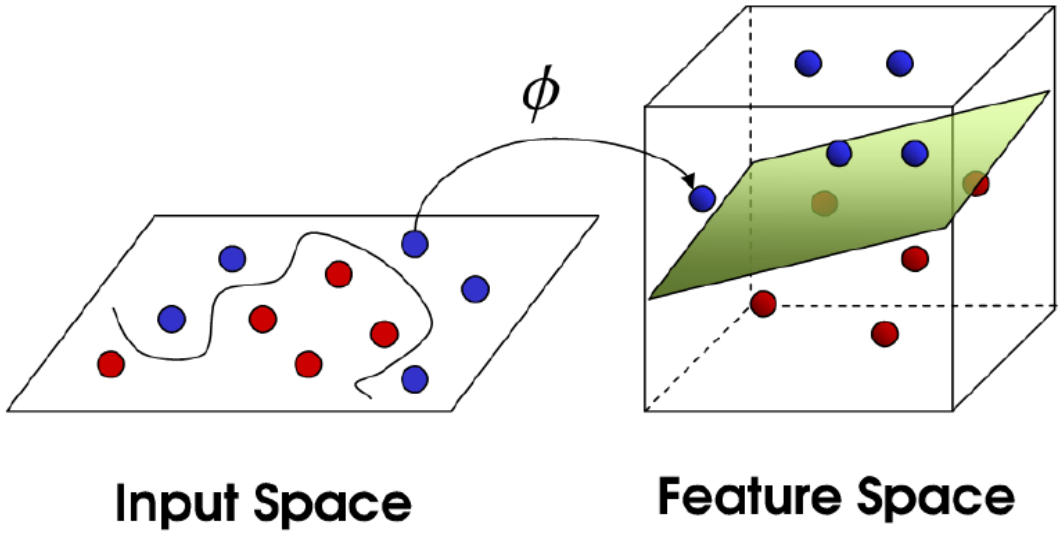
머신러닝 – Support Vector Machine



- 머신러닝의 SVM: POSITIVE 데이터 세트와 NEGATIVE 데이터 세트를 나누어 그 사이의 가장 넓은 공간을 기점으로 분류를 하는 지도학습의 방식

- 데이터 시트가 많을수록, 서로 다른 데이터 시트의 거리가 멀수록 좋은 ACCURACY를 얻을 수 있음

머신러닝 – Support Vector Machine



- "특징 공간(FEATURE SPACE)" : SVM이 DECISION BOUNDARY와 커널을 통해 분류 하는 공간

- 특징 공간으로 이송된 데이터 세트는 더욱 확연하게 구분되어 정확도를 높이는 방안이 됨

머신러닝 – Training

Positive Data Set

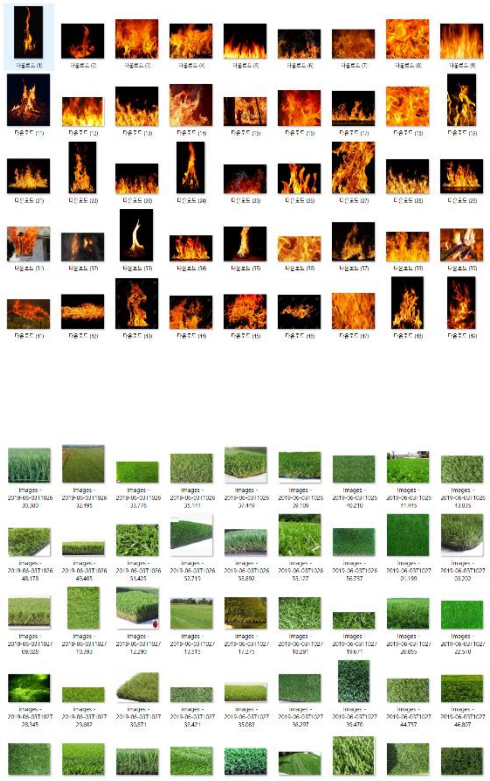
2,560장의 불, 화재 사진 학습

Negative Data Set

사람, 건물, 자연 등이 포함된 900장의 사진 학습

03. 연구내용

머신러닝 – Training



```
In [2]: import numpy as np

        np.random.seed(3)

        from keras.preprocessing.image import ImageDataGenerator

        train_datagen = ImageDataGenerator(rescale = 1.0/255)
        train_generator = train_datagen.flow_from_directory(
            directory = r"./TRAIN_DIR/",
            target_size = (224, 224),
            color_mode = "rgb",
            batch_size = 50,
            class_mode = 'binary'
        )

        test_datagen = ImageDataGenerator(rescale = 1.0/255)
        test_generator = test_datagen.flow_from_directory(
            directory = r"./TEST_DIR/",
            target_size = (224, 224),
            color_mode = "rgb",
            batch_size = 32,
            class_mode = 'binary'
        )
```

```
Using TensorFlow backend.

Found 3480 images belonging to 2 classes.
Found 474 images belonging to 2 classes.
```

```
In [3]: import cv2
        import matplotlib.pyplot as plt
        %matplotlib inline
```

```
In [4]: for i in range(5):
            filename = "TRAIN_DIR/fire/fire (%d).jpg" % (i+1)
            img = cv2.imread(filename)
            plt.subplot(1, 5, i+1)
            plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
            plt.axis('off')
```

```
In [5]: win_size = (48, 96)
        block_size = (16, 16)
        block_stride = (8, 8)
        cell_size = (8, 8)
        num_bins = 9

        HOG = cv2.HOGDescriptor(win_size, block_size, block_stride, cell_size, num_bins)
```

```
In [6]: import numpy as np
        import random

        random.seed(42)

        X_pos = []
        for i in random.sample(range(2560), 500):
            filename = "TRAIN_DIR/fire/fire (%d).jpg" % (i+1)
            img = cv2.imread(filename)
            img = cv2.resize(img, (64, 64))

            if img is None:
                print('Could not find image named %s' % filename)
```

각각의 데이터 세트를 받아 머신 러닝 실시

머신러닝 – Training

```
In [7]:  X_pos = np.array(X_pos, dtype = np.float32)
         Y_pos = np.ones(X_pos.shape[0], dtype = np.int32)
         X_pos.shape, Y_pos.shape
Out[7]:  ((500, 1980, 1), (500,))

In [8]:  negdir = "TRAIN_DIR/not_fire"

In [10]: import os

         hroi = 128
         wroi = 64
         X_neg = []
         for negfile in os.listdir(negdir):
             filename = "%s/%s" % (negdir, negfile)
             img = cv2.imread(filename)
             img = cv2.resize(img, (256, 256))

             for j in range(5):
                 rand_y = random.randint(0, img.shape[0] - hroi)
                 rand_x = random.randint(0, img.shape[1] - wroi)
                 roi = img[rand_y:rand_y + hroi, rand_x:rand_x + wroi, :]
                 X_neg.append(HOG.compute(roi, (64, 64)))

In [11]: X_neg = np.array(X_neg, dtype = np.float32)
         Y_neg = -np.ones(X_neg.shape[0], dtype = np.int32)
         X_neg.shape, Y_neg.shape
Out[11]: ((4500, 1980, 1), (4500,))
```

```
In [30]: SVM = train_svm(X_train, Y_train)

In [33]: score_svm(SVM, X_train, Y_train)
Out[33]: 1.0

In [34]: score_svm(SVM, X_test, Y_test)
Out[34]: 0.892
```

머신 러닝 데이터 세트 훈련 결과: ACCURACY 89.2%

하드웨어

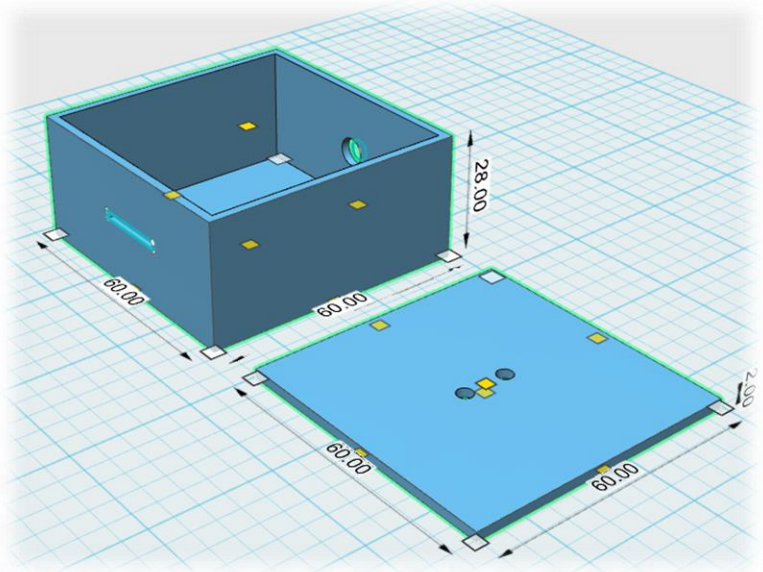
Hardware

HardWare

결과물을 지탱하고 감쌀 하드웨어에 대한
123D Design 3D 설계에 대한 내용

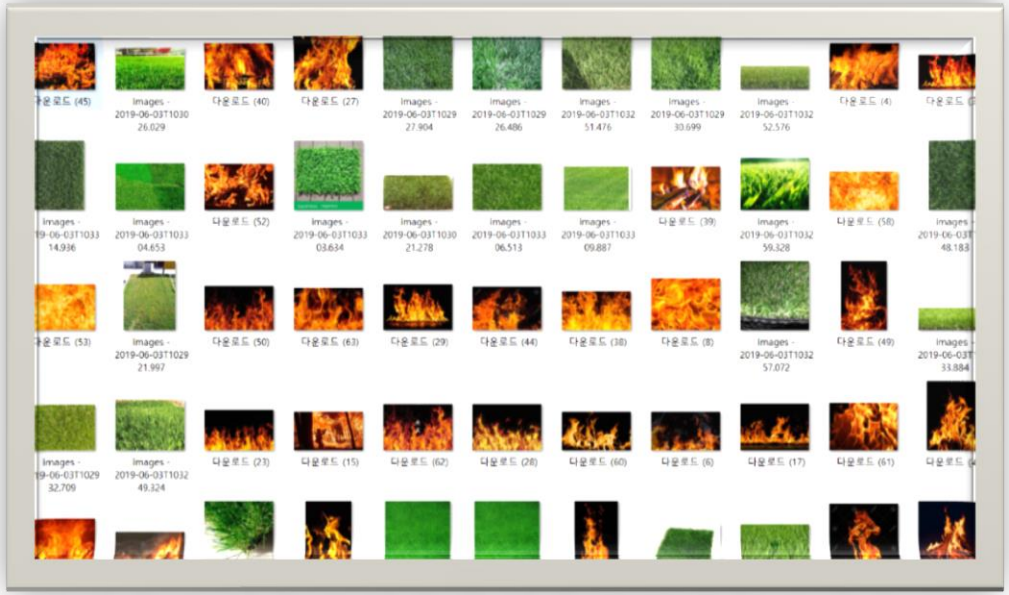
- 123D Design를 이용한 설계

하드웨어 – 123D Design을 이용한 설계



- 영상처리를 위한 카메라와 LED등을 위한 하드웨어 설계
- 출력은 3D 프린터를 사용하였다. FDM 방식을 사용하였고 사용소재는 PLA를 사용
- PLA 소재는 가볍고 가격이 저렴하며, 내구성이 튼튼하다는 장점
- 녹는점은 215℃로 화재 탐지기의 외관으로는 적합하지 않은 소재이나, 시제품용으로 장점이 우수하여 사용
- 실제 화재탐지기 외관으로는 방화문의 소재로 사용되는 열연전기아연도금강판 (HOT-EGI), 유니텍스(UNITEX) 등이 적합

결 론 – Accuracy 부분

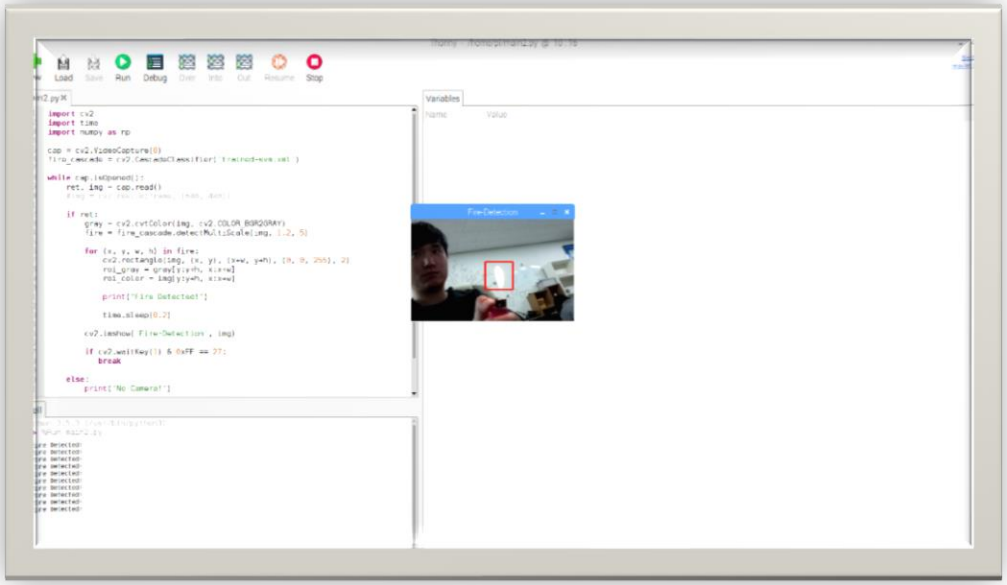


Random() 함수

- 윈도우10 환경에서는 Accuracy 89.2%
- Raspbian Stretch 환경에서는 Accuracy 88.9%
- Accuracy 85%가 넘는 우수한 결과로 매우 화재 검출 적중률 높은 머신러닝 모델임을 확인

변동오차 ±0.25%

결 론 – 화재 감지 및 검출



영상 스트리밍

- 라이터 점화 이후, 즉시 화염 부분에 빨간색의 사각형 테두리가 그려지며 "FIRE DETECTED" 메시지 출력
- 인체나 태양광, 조도, 그림자의 변화에도 오검출 하는 경우 없음
- 이때 CPU 사용량은 20% ~ 50% 사이이며 3분 이상 라이터를 점화하여 불을 검출 가능

소스코드 및 화재 검출

결 론 – 다른 논문과의 비교

화재검출 알고리즘	기존 시스템	딥러닝 CNN 이용	가우시안 필터 이용	머신러닝 CVM 이용
경제성	△	X	O	O
Accuracy	△	O	X	O

- 딥러닝 : 소모 전력이 매우 크며 높은 성능을 요구
- 가우시안 필터 : 열 배출이 많은 에어컨을 화재로 오판하는 등의 적중률 문제점이 존재
- 머신러닝 : ACCURACY 89.2%로 특징 검출을 통해 높은 화재 적중률을 보이며 화재 이외의 인체나 태양광에 오검출을 하는 일이 없어 적합

구 현

[Inha University]
12131334 백동원 | 12141466 정선우 | 12163651 허정인

인하대학교 전기종합설계 Project

SVM 알고리즘을 통한 화재 감지 시스템


인하대학교
1954
INHA UNIVERSITY