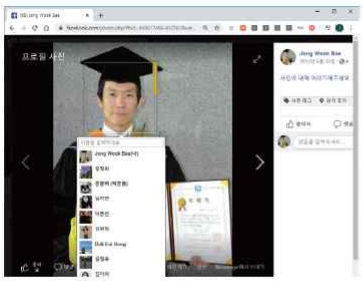
CHAPTER 11
영상 처리 응용 사례 I

11.2 HAAR 분류기를 이용한 얼굴검출 및 성별 분류

1

11.2 Haar 분류기를 이용한 얼굴검출 및 성별 분류

- 얼굴 검출 및 인식 응용
  - 저가형 디지털 카메라에서도 얼굴 검출 후 액정에 표시
  - 페이스북이나 구글에 사진을 올리면 얼굴 영역 검출



참조: https://cdnimg.rg.ru

〈그림 11.2.1〉 얼굴 검출 응용 예

2

11.2.1 Haar 기반 분류기

- **Haar 특징**
  - Viola and Jones가 제안(2001)
  - 논문 “Rapid Object Detection Using a Boosted Cascade of Simple Features”
  - 얼굴과 얼굴이 아닌 것의 차이를 효율적으로 보여줄 수 있는 Haar 유사 특징(Haar-like features)을 이용한 방법 제안
  - 원리:
    - Feature(특징) 선택 : 중요한 features 선택
    - Attention 집중 : 잠재적 영역에 집중
    - 빠른 feature 평가를 위해 적분(integral) 영상 사용

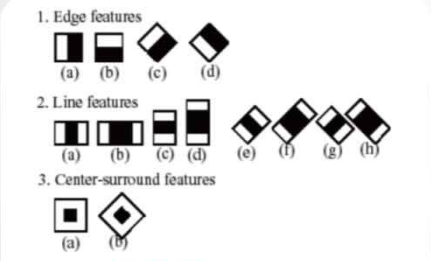
3

11.2.1 Haar 기반 분류기

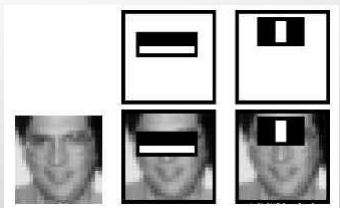
- **Haar 특징**

① 중요한 features => Haar features

- 특정 Feature는 명암의 차가 있다는 이론
- 이미지에서 영역과 영역의 밝기차를 이용하여 Feature로서의 특징을 찾아내는 방법
- 사람의 얼굴, 눈, 코, 입 등은 특징적인 밝기 차가 있기에 이를 통해 Feature를 찾아냄
- Feature값은 흰색 영역의 화소값 합과 검은색 영역의 화소값 합의 차로 정의



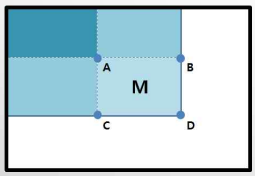
〈그림 11.2.2〉 유사 하르 특징



4

11.2.1 Haar 기반 분류기

- **Haar 특징**
  - **② 적분 영상(Integral image)**
    - 영상에는 무수히 많은 영역이 존재
      - 24×24픽셀의 영상에 5가지의 Haar 필터를 적용한다면 영역의 수는 180,000개에 달함
    - Haar 필터를 계산할 때마다 이러한 영역의 픽셀 수를 일일이 계산한다면, 연산 시간이 상당히 오래 걸리므로 각 점에서의 적분 영역을 미리 계산해 두어 연산 시간을 줄임.

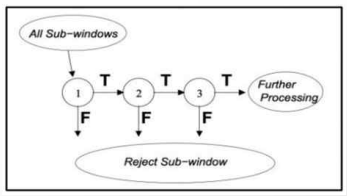


      - A 점은 영상의 왼쪽 위부터 A 점까지의 화소의 합. B,C,D 역시 같음.
      - 영역 M의 면적 = **D - B - C + A**
    - 적분 영상은 매번 수행해야 하는 여러 화소 합산에 사용되므로, 사각 영역의 합계, 평균 등을 빠르게 계산할 수 있다.

5

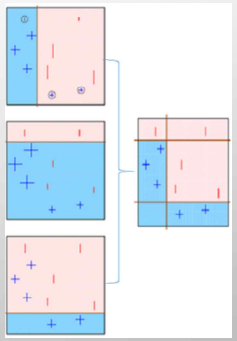
11.2.1 Haar 기반 분류기

- **Haar 특징**
  - **③ Haar Cascades 분류기(classifier)**
    - 여러 개의 분류기를 순차적으로 사용



    - 처음에 간단한 분류기를 적용하고, 진행할수록 복잡한 분류기 적용
      - 어려운 작업을 수행하기 위해 모든 간단한 분류기들 결합
    - 단순 분류기를 통과한 후보에만 시간이 많이 걸리는 강력한 분류기가 적용

      → 검출 속도 크게 향상



6

11.2.1 Haar 기반 분류기

- OpenCV의 Cascades 분류기
  - 1,000개 이상의 얼굴 영상과 10,000개 이상의 얼굴이 아닌 영상을 사용하여 학습
  - https://github.com/opencv/opencv/tree/master/data/haarcascades
  - https://github.com/AlexeyAB/OpenCV-detection-models/tree/master/haarcascades

〈표 11.2.1〉 XML 검출기 목록

cascade classifer 분류기	XML 파일명
Face detector (default)	haarcascade_frontalface_default.xml
Face detector (fast Harr)	haarcascade_frontalface_alt2.xml
Face detector (fast LBP)	lbpcascade_frontalface.xml
Eye detector	haarcascade_eye.xml
Eye detector (separate for left)	haarcascade_lefteye_2splits.xml
Eye detector (separate for right)	haarcascade_righteye_2splits.xml
Mouth detector	haarcascade_mcs_mouth.xml
Nose detector	haarcascade_mcs_nose.xml
Whole person detector	haarcascade_fullbody.xml

7

11.2.1 Haar 기반 분류기

- OpenCV의 Cascades 분류기
  - ➢ `cv2.CascadeClassifier([filename]) → <CascadeClassifier object>`
    - `filename` – `classifier` 파일이름

함수 인수와 반환자료형 구조	
void CascadeClassifier::detectMultiScale(InputArray image, CV_OUT vector<Rect>& objects, double scaleFactor = 1.1, int minNeighbors = 3, int flags = 0, Size minSize = Size(), Size maxSize = Size());	
인수	설명
InputArray image	객체 검출 대상 행렬 (8비트 명암도 영상)
vector<Rect>& objects	반환되는 검출 객체 사각형
double scaleFactor	영상 크기 감소에 대한 규정
int minNeighbors	이웃 후보 사각형의 개수
int flags = 0	과거 함수에서 사용하던 flag
Size minSize	가능한 객체 최소 크기 – 이보다 작은 객체 무시
Size maxSize	가능한 객체 최대 크기 – 이보다 큰 객체 무시

다양한 크기의 얼굴 검출 가능

8

Haar 특징을 이용한 Face detection

```
def preprocessing(img):  # 검출 전처리
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)  # 명암도 영상 변환
    gray = cv2.equalizeHist(gray)  # 히스토그램 평활화
    return gray
```

Common 폴더 haar_utils.py에 저장

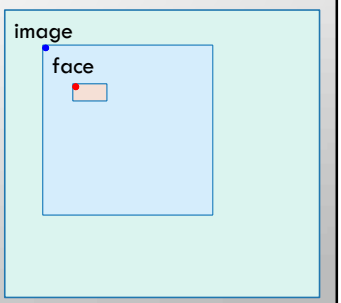
```
from Common.haar_utils import *
```

```
image = cv2.imread('images/people.jpg', cv2.IMREAD_COLOR)
if image is None: raise Exception("영상 파일 읽기 에러")
img_gray = preprocessing(image)

# Face 분류기 로드
face_cascade = cv2.CascadeClassifier("haarcascade/haarcascade_frontalface_alt2.xml")  # 정면 검출기
if face_cascade.empty(): raise IOError('Unable to load the face cascade classifier xml file')

faces = face_cascade.detectMultiScale(img_gray, 1.1, 2, 0, (100, 100))  # 얼굴 검출

for face in faces :                              # for x, y, w, h in faces:
    x, y, w, h = face
    cv2.rectangle(image, face, (255, 0, 0), 2) # cv2.rectangle(image, (x,y,w,h), (255, 0, 0), 2)

cv2.imshow("Face Detector", image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

9

Haar 특징을 이용한 Eye detection

```
# Eye 분류기 로드
eye_cascade = cv2.CascadeClassifier('haarcascade/haarcascade_eye.xml') # 눈 검출기
if eye_cascade.empty(): raise IOError('Unable to load the eye cascade classifier xml file')

# 검출된 얼굴에서 눈 검출
for face in faces :
    x, y, w, h = face
    face_image = img_gray[y:y + h, x:x + w]  # 얼굴 영역 영상 가져오기

    eyes = eye_cascade.detectMultiScale(face_image, 1.15, 7, 0, (25, 20))  # 눈 검출 수행
    if len(eyes) == 2:  # 눈 사각형이 검출되면
        for ex, ey, ew, eh in eyes:
            center = (x + ex + ew // 2, y + ey + eh // 2)
            cv2.circle(image, center, 10, (0, 255, 0), 2)  # 눈 중심에 원 그리기
    else: print("눈 미검출")

    cv2.rectangle(image, face, (255, 0, 0), 2)  # 얼굴 검출 사각형 그리기

cv2.imshow("Eye Detector", image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```



10

Haar 특징을 이용한 Mouth detection

```
# Mouth 분류기 로드
mouth_cascade = cv2.CascadeClassifier('haarcascade/haarcascade_mcs_mouth.xml') # 입 검출기
if mouth_cascade.empty(): raise IOError('Unable to load the mouth cascade classifier xml file')

# 검출된 얼굴에서 입 검출
for face in faces :
    x, y, w, h = face
    face_image = img_gray[y:y + h, x:x + w]  # 얼굴 영역 영상 가져오기

    mouths = mouth_cascade.detectMultiScale(face_image, 1.7, 11)  # 입 검출 수행

    for mouth in mouths:
        mx, my, mw, mh = mouth
        cv2.rectangle(image, (x+mx, y+my, mw, mh, (0, 255, 0), 3)

    cv2.rectangle(image, face, (255, 0, 0), 2)  # 얼굴 검출 사각형 그리기

cv2.imshow("Eye Detector", image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

11

CVLIB을 이용한 Face detection

- **cvlib 란?**
  - https://www.cvlib.net/
  - 파이썬에서 얼굴, 객체 인식을 위한 사용하기 쉬운 라이브러리
  - cvlib 설치 in PyCharm
    - opencv와 tensorflow를 사용하고 있기 때문에, 함께 설치해야 함
    - 메뉴 File > Settings > Project > Python Interpreter에서 cvlib과 tensorflow 추가(opencv-python은 이미 설치)

```
import cv2
import cvlib as cv

image = cv2.imread('images/people2.jpg', cv2.IMREAD_COLOR)
if image is None: raise Exception("영상 파일 읽기 에러")

# CVLIB를 이용한 얼굴 찾기
faces, confidences = cv.detect_face(image)

for (x, y, x2, y2), conf in zip(faces, confidences):
    cv2.putText(image, str(conf), (x,y-10), cv2.FONT_HERSHEY_PLAIN, 1, (0, 255, 0), 1) # 확률 출력하기
    cv2.rectangle(image, (x, y), (x2, y2), (0, 255, 0), 2) # 얼굴위치 bbox 그리기

cv2.imshow('CVlib - Face detector', image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

12

CVLIB을 이용한 성별 분류(gender classification)

```
# 얼굴 찾기
faces, confidences = cv.detect_face(image)

for (x, y, x2, y2)in faces:
    face_img = image[y:y2, x:x2]

    # CVLIB를 이용한 성별 분류
    label, confidence = cv.detect_gender(face_img)

    cv2.rectangle(image, (x, y), (x2, y2), (0, 255, 0), 2)

    gender = np.argmax(confidence)
    text = f'{label[gender]}:{confidence[gender]:.1%}'
    cv2.putText(image, text, (x,y-10), cv2.FONT_HERSHEY_PLAIN, 1, (0, 0, 255), 1)

cv2.imshow('CVlib - Gender classification', image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

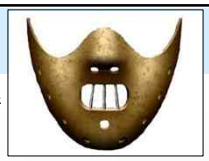
13

Play with face

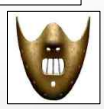
```
#가면 영상
face_mask = cv2.imread('ano_inv.png')
h_mask, w_mask = face_mask.shape[:2]

#검출된 얼굴에 가면 합성
for face in faces:
    x, y, w, h = face
  # 검출된 얼굴에 대한 사각형 영역을 관심 영역(ROI)로 설정
   if h > 0 and w > 0:
      frame_roi = frame[y:y+h, x:x+w]
   # 가면 영상의 크기를 검출된 얼굴의 크기와 같도록 resize
      face_mask_small=cv2.resize(face_mask,(w,h),interpolation=cv2.INTER_AREA)
   # ① 가면 마스크에서 검지 않은 부분만 통과(검은색 부분은 투명)
      gray_mask = cv2.cvtColor(face_mask_small, cv2.COLOR_BGR2GRAY)
      ret, mask = cv2.threshold(gray_mask, 50, 255, cv2.THRESH_BINARY_INV)
      masked_face = cv2.bitwise_and(face_mask_small,face_mask_small,mask=mask)
   # ② 검출된 얼굴에서 가면 마스크의 검은 부분만 통과(검지 않은 부분은 투명)
      mask_inv = cv2.bitwise_not(mask)
      masked_frame = cv2.bitwise_and(frame_roi, frame_roi, mask=mask_inv)
   # 흰색 마스크와 마스크된 얼굴 합성 (① + ②)
      frame[y:y+h, x:x+w] = cv2.add(masked_face, masked_frame)
```

face_mask의 배경이 흰 경우

face_mask



frame_roi



face_mask_small

mask

masked_face

mask_inv

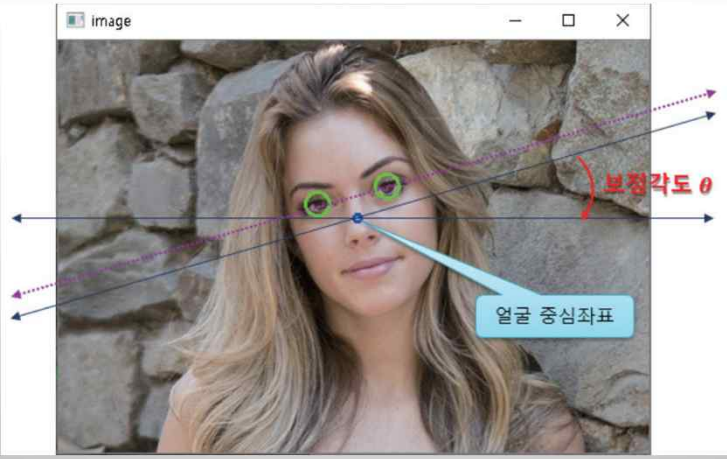
masked_frame

frame

14

11.2.4 얼굴 기울기 계산 및 보정

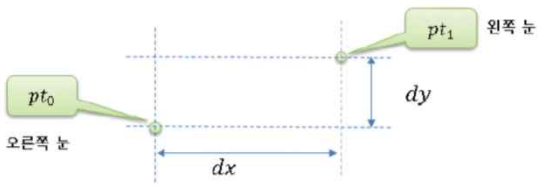
- 얼굴의 기울어진 각도는 두 눈의 중심좌표 이용
  - 눈 검출 필요 , 중심점에서 기울기만큼 회전



15

11.2.4 얼굴 기울기 계산 및 보정

- 두 눈 좌표의 차분 좌표으로 역탄젠트를 취하면 기울기 계산 가능



$$(dx,\ dy) = pt_1 - pt_0$$

$$\theta = \tan^{-1}\left(\frac{dy}{dx}\right)$$

〈그림 11.2.6〉 검출된 두 눈을 이용한 기울기 보정 방법

16

11.2.4 얼굴 기울기 계산 및 보정

```
def correct_image(image, face_center, eye_centers):
    pt0, pt1 = eye_centers                                  # 좌, 우 눈 중심 좌표
    if pt0[0] > pt1[0]: pt0, pt1 = pt1, pt0                 # 좌, 우 눈 위치 맞바꿈

    dx, dy = np.subtract(pt1, pt0)                          # 두 좌표간 차분 계산
    angle = cv2.fastAtan2(dy, dx)                           # 차분으로 기울기 계산
    rot_mat = cv2.getRotationMatrix2D(face_center, angle, 1)

    size = image.shape[1::-1]                               # 행태와 크기는 역순
    corr_image = cv2.warpAffine(image, rot_mat, size, cv2.INTER_CUBIC)

    eye_centers = np.expand_dims(eye_centers, axis=0)       # 눈 좌표 차원 증가

    corr_centers = cv2.transform(eye_centers, rot_mat)      # 어파인(회전) 변환 좌표 계산
    corr_centers = np.squeeze(corr_centers, axis=0)         # 보정 좌표 차원 감소

    return corr_image, corr_centers                         # 보정 결과 반환
```

```
for face in faces :
    x, y, w, h = face
    face_image = image[y:y + h, x:x + w]  # 얼굴 영역 영상 가져오기
    eyes = eye_cascade.detectMultiScale(face_image, 1.15, 7, 0, (25, 20))  # 눈 검출 수행
    if len(eyes) == 2:  # 눈 사각형이 검출되면
        face_center = (x + w // 2, y + h // 2)
        eye_centers = [(x + ex + ew // 2, y + ey + eh // 2) for ex, ey, ew, eh in eyes]
        corr_image, corr_center = correct_image(image, face_center, eye_centers)  # 기울기 보정
        cv2.circle(corr_image, tuple(corr_center[0]), 5, (0, 255, 0), 2)
        cv2.circle(corr_image, tuple(corr_center[1]), 5, (0, 255, 0), 2)
        cv2.circle(corr_image, face_center, 3, (0, 0, 255), 2)
        cv2.imshow("correct_image", corr_image)
```

17

단원 요약

- 6. 하르 기반 캐스케이드 분류기는 단순한 여러 개의 분류기를 순차적으로 사용하여 분류를 수행한다. 처음에 간단한 검출기를 적용하고, 진행할수록 복잡한 검출기를 적용한다. 단순 검출기를 통과한 후보에만 시간이 많이 걸리는 강력한 검출기가 적용되기 때문에 검출 속도를 빨리할 수 있다.

- 7. OpenCV에서 하르 기반 검출기는 얼굴, 눈, 코, 입 등을 검출하는 XML 파일들이 있다. 객체 검출을 위해서는 CascadeClassifier 클래스를 이용하며, 내부 메서드인 CascadeClassifier.load() 함수로 해당 XML 파일을 로드하고, CascadeClassifier. detectMultiScale() 함수로 검출을 수행한다.

- 8. 하르 검출기로 검출된 얼굴과 눈 영역으로 얼굴 중심좌표와 눈 중심좌표를 구할 수 있다. 그리고 두 눈의 중심좌표로 얼굴의 기울어진 각도를 계산할 수 있으며, 이 각도만큼 얼굴 중심좌표를 기준으로 회전하면 기울기를 보정할 수 있다.

18