영상 처리 응용 사례

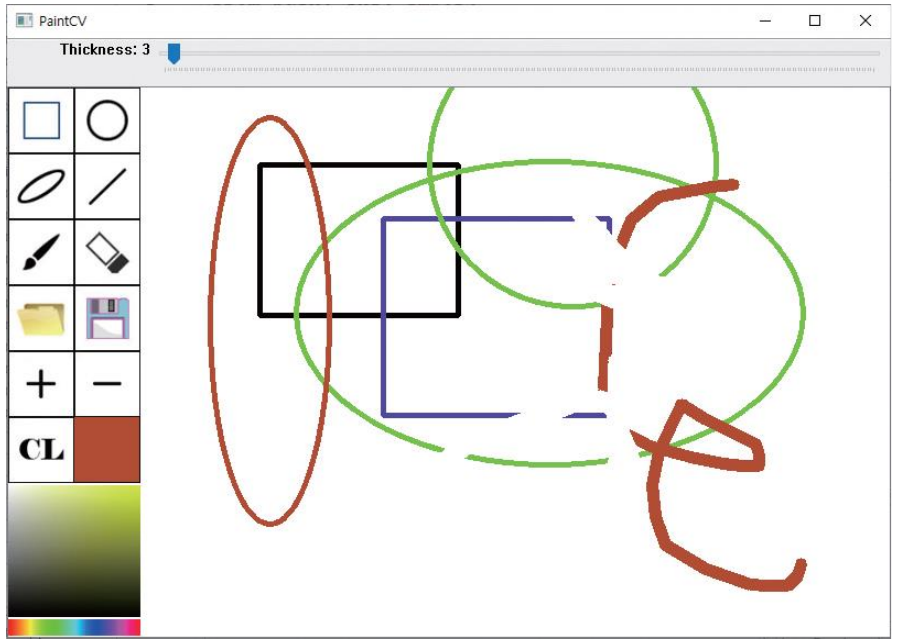
담당교수: 김민기

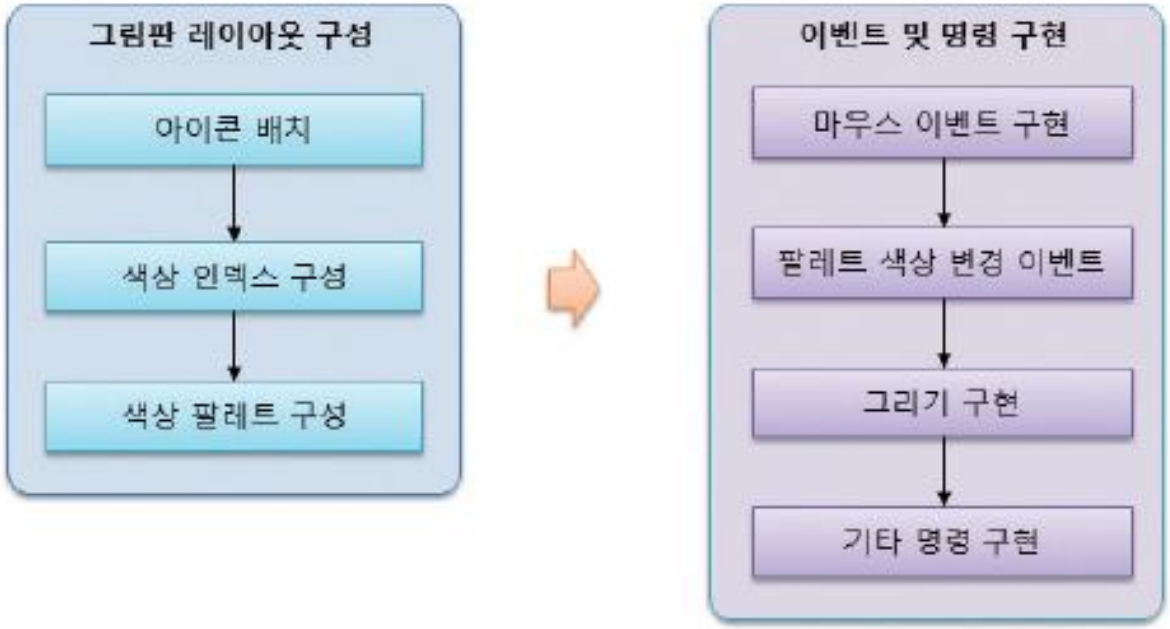
Contents

1. 그림판 프로그램
2. 얼굴 검출 및 성별 분류

1. 그림판 프로그램

❖OpenCV API를 활용해서 그림판 프로그램을 구현해 보자.

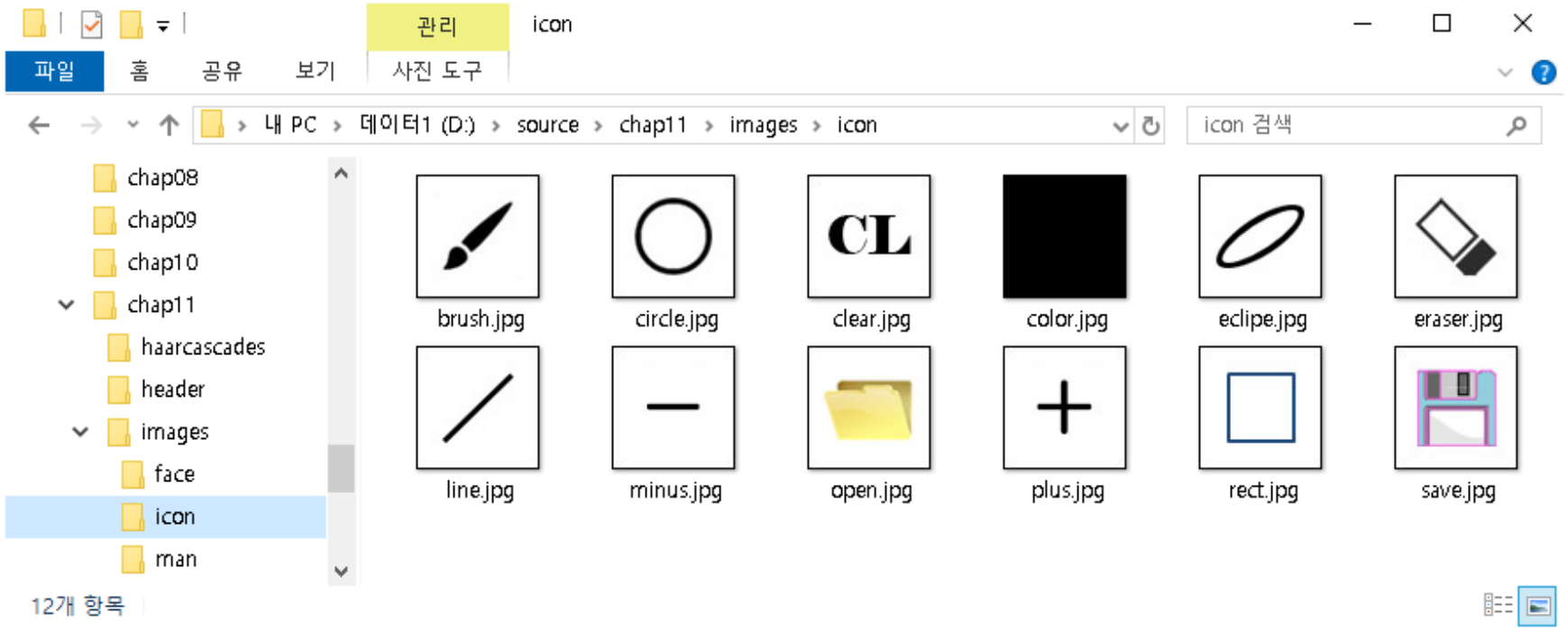




〈그림 11.1.2〉 그림판 프로그램 전체 구현 과정

아이콘 배치

❖버튼 아이콘 파일: ./image/icon 폴더



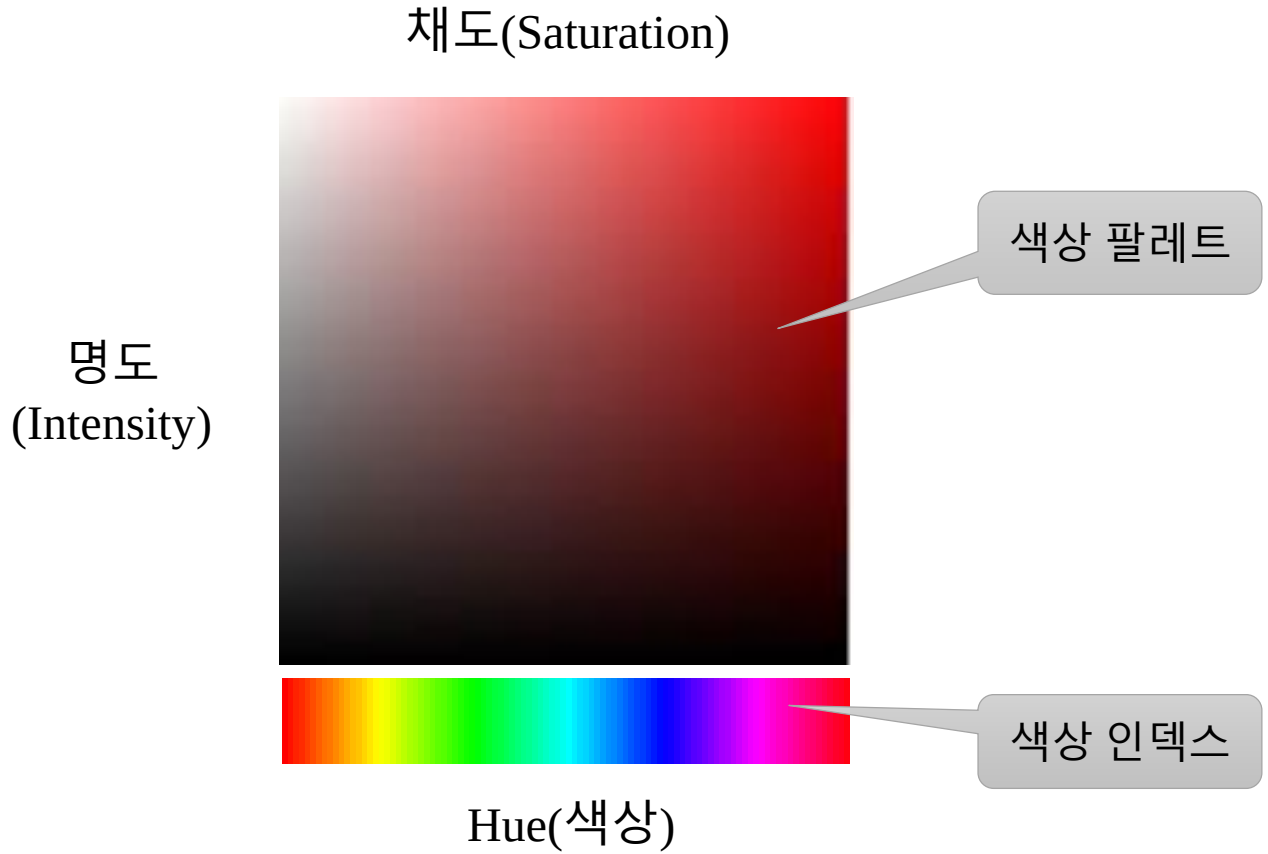
예제 11.1.1 아이콘 레이아웃 구성 01.place_icons.py

```
01  import numpy as np, cv2
02
03  def place_icons(image, size):
04      icon_name = ["rect" , "circle", "eclipe", "line", "brush", "eraser", # 아이콘 파일 이름
05                   "open", "save", "plus", "minus", "clear" , "color"]
06      icons = [(i%2, i//2, 1, 1) for i in range(len(icon_name))]
07      icons = np.multiply(icons, size*2)                  # icons 모든 원소에 size 곱함
08      # 아이콘들의 위치(roi=x, y, w, h)
09      for roi, name in zip(icons, icon_name):
10          icon = cv2.imread("images/icon/%s.jpg" %name , cv2.IMREAD_COLOR)
11          if icon is None: continue                       # 영상파일 없으면 다음 파일 읽기
12          x, y, w, h = roi                                # 아이콘 영역
13          image[y:y+h, x:x+w] = cv2.resize(icon, size)    # 영상의 아이콘 영역에 이미지 복사
14      return list(icons)
15
16  image = np.full((500, 800, 3), 255, np.uint8)
17  icons = place_icons(image, (60, 60))                    # 아이콘 배치, 아이콘 크기
18  cv2.imshow("PaintCV", image)
19  cv2.waitKey(0)
```

색상 인덱스와 색상 팔레트 구현

❖컬러 색상을 표현하려면 3차원 색 공간 필요

- RGB모델 → 3개 채널의 독립성으로 인해서 원하는 색상 찾기 힘듦
- 색상의 명확한 표현을 위해서 HSV 모델로 표현



❖ 색상 인덱스 영역 그리기

색상 인덱스 영역

```
def create_hueIndex(image, roi):
    x, y, w, h = roi                                              # 관심 영역 너비, 높이
    index = [[(j, 1, 1) for j in range(w)] for i in range(h)]     # 가로 방향 좌표 행렬
    ratios = (180/w, 255, 255)      # Hue(0~180), Saturation(0~255), Value(0~255)
    hueIndex = np.multiply(index, ratios).astype('uint8')         # HSV 화소값 행렬
    image[y:y+h, x:x+w] = cv2.cvtColor(hueIndex, cv2.COLOR_HSV2BGR)
```

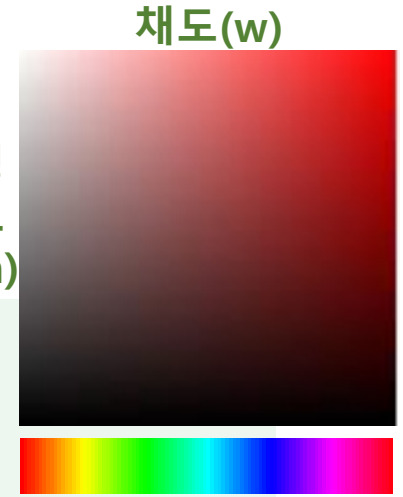
색상 : 0~180

색상 인덱스 행렬 구성

그림판 영상의 관심 영역에 색상 인덱스 복사

HSV –> BGR 변환

❖ 팔레트 영역 그리기



팔레트 영역

```
def create_colorPlatte(image, idx, roi):
    x, y, w, h = roi
    hue = idx-x                                              # 색상값 – 클릭 x 좌표
    palette = [[(hue, j, h-i-1) for j in range(w)] for i in range(h)]     # 팔레트 좌표 생성

    ratios = (180/w, 255/w, 255/h)                           # 팔레트 크기에 따른 화소비율
    palette = np.multiply(palette, ratios).astype('uint8')   # 팔레트 좌표와 비율 곱

    image[y:y+h, x:x+w] = cv2.cvtColor(palette, cv2.COLOR_HSV2BGR)
```

그림판 영상의 관심 영역에
색상 팔레트 복사

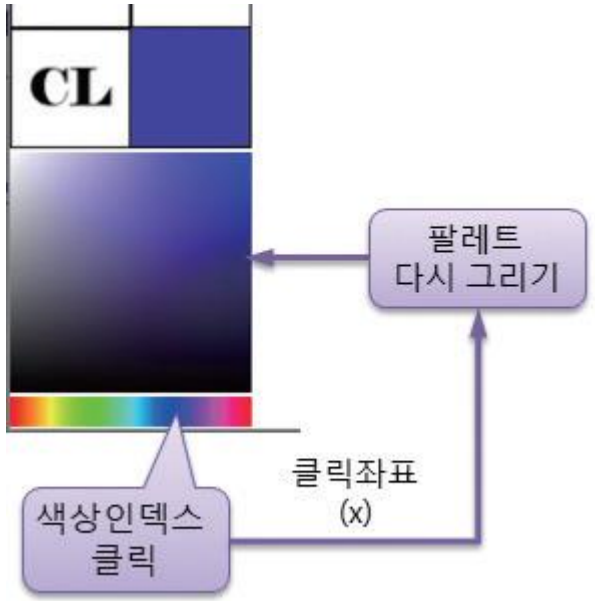
예제 11.1.2 그림판 아이콘 배치 – 02.event_platte.py

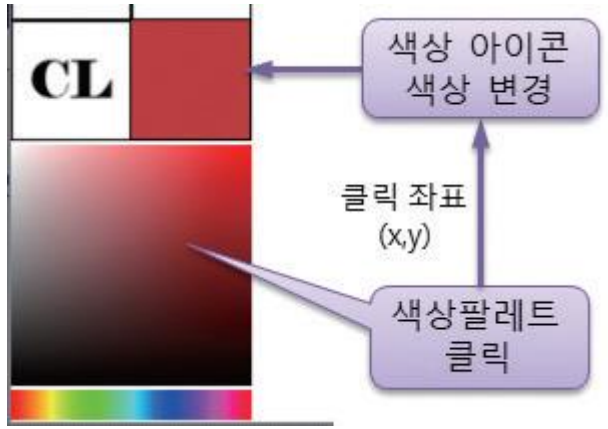
```
01  from paint_init import *                    # 상수, 전역변수 임포
02  from paint_utils import *                   # 메뉴, 팔레트 생성
03
04  image = np.full((500, 800, 3), 255, np.uint8)
05  icons = place_icons(image, (60, 60))        # 아이콘 배치, 아이콘
06  x, y, w, h = icons[-1]                      # 아이콘 사각형 마지
07
08  PALETTE_roi = (0, y+h+2, 120, 120)          # 색상 팔레트 roi
09  hueIndex_roi = (0, y+h+124 , 120, 15)       # 색상 인덱스 roi
10  icons.append(PALETTE_roi)                   # icons 리스트에 색상 팔레트 추가
11  icons.append(hueIndex_roi)                  # icons 리스트에 색상 인덱스 추가
12  create_colorPlatte(image, 0, icons[PALETTE])  # 색상 팔레트 생성
13  create_hueIndex(image, icons[HUE_IDX])      # 색상 인덱스 생성
14
15  cv2.imshow("PaintCV", image)
16  cv2.waitKey(0)
```

마우스 이벤트 구현

❖이벤트 발생시 색상 변경 하기

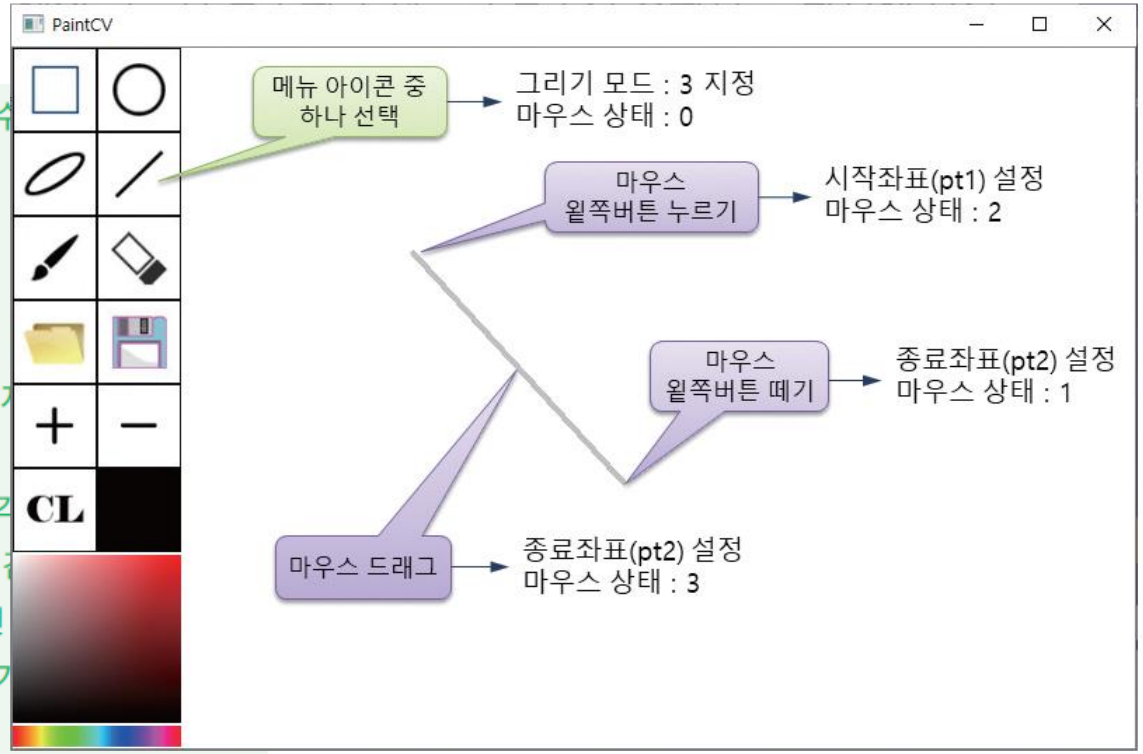
- 색상 인덱스 클릭 → 색상 인덱스 색상 기준으로 팔레트 다시 그리기
- 팔레트 클릭 → 현재 색상 아이콘 색 변경





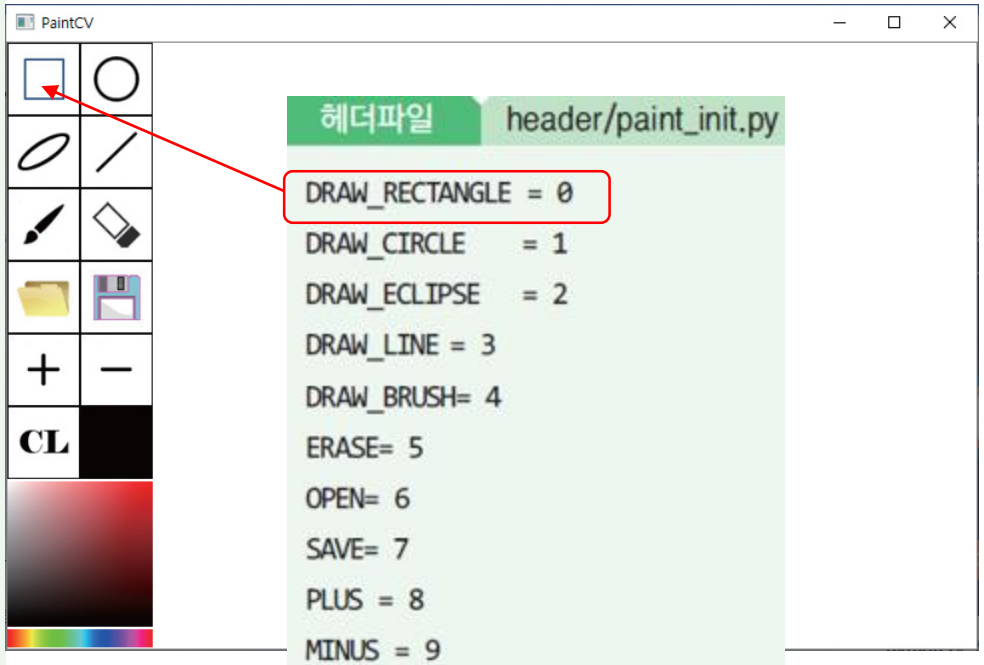
❖ 마우스 이벤트 구현

```
01  def onMouse(event, x, y, flags, param):                    # 마우스 콜백 함수
02      global pt1, pt2, mouse_mode, draw_mode
03
04      if event == cv2.EVENT_LBUTTONUP:                       # 왼쪽 버튼 떼기
05          pt2 = (x, y)                                       # 종료 좌표 저장
06          mouse_mode = 1                                     # 버튼 떼기 상태 지
07
08          for i, (x0, y0, w, h) in enumerate(icons):         # 메뉴 아이콘 사각
09              if x0 <= x < x0+w and y0 <= y < y0+h:          # 메뉴 클릭 여부 검
10                  if i < 6:                                  # 그리기 명령이면
11                      mouse_mode = 0                         # 마우스 상태 초기
12                      draw_mode = i                          # 그리기 모드
13                  else: command(i)                           # 일반 명령 수행
14                  return
15
16      elif event == cv2.EVENT_LBUTTONDOWN:                   # 왼쪽 버튼 누르기
17          pt1 = (x, y)                                       # 시작 좌표 저장
18          mouse_mode = 2
19
20      if mouse_mode >= 2:                                    # 왼쪽 버튼 누르기 또는 드래그
21          mouse_mode = 0 if x < 125 else 3                   # 메뉴 영역 확인– 마우스 상태 지정
22          pt2 = (x, y)
```



mouse_mode	의미
0	동작 종료, 초기화
1	마우스 왼쪽 버튼 떼기
2	마우스 왼쪽 버튼 누르기
3	마우스 드래그

```python
def draw(image, color=(200, 200, 200)):
    global draw_mode, thickness, pt1, pt2              # 전역 변수 사용

    if draw_mode == DRAW_RECTANGLE:                    # 사각형 그리기
        cv2.rectangle(image, pt1, pt2, color, thickness)

    elif draw_mode == DRAW_LINE:                       # 직선 그리기
        cv2.line(image, pt1, pt2, color, thickness)

    elif draw_mode == DRAW_BRUSH:                      # 브러시 그리기
        cv2.line(image, pt1, pt2, color, thickness * 3)
        pt1 = pt2                                      # 종료 좌표를 시작 좌표로 지정

    elif draw_mode == ERASE:                           # 지우개
        cv2.line(image, pt1, pt2, (255, 255, 255), thickness * 5)
        pt1 = pt2

    elif draw_mode == DRAW_CIRCLE:                     # 원 그리기
        d = np.subtract(pt1, pt2)                      # 두 좌표 차분
        radius = int(np.sqrt(d[0] ** 2 + d[1] ** 2))   # 피타고라스 정리로 거리측정
        cv2.circle(image, pt1, radius, color, thickness)

    elif draw_mode == DRAW_ECLIPSE:                    # 타원 그리기
        center = np.abs(np.add(pt1, pt2)) // 2         # 두 좌표의 중심점 구하기
        size = np.abs(np.subtract(pt1, pt2)) // 2      # 두 좌표의 크기의 절반
        cv2.ellipse(image, tuple(center), tuple(size), 0, 0, 360, color, thickness)

cv2.imshow("PaintCV", image)
```



헤더파일 header/paint_init.py

```
DRAW_RECTANGLE = 0
DRAW_CIRCLE    = 1
DRAW_ECLIPSE   = 2
DRAW_LINE = 3
DRAW_BRUSH= 4
ERASE= 5
OPEN= 6
SAVE= 7
PLUS = 8
MINUS = 9
CREAR = 10
COLOR = 11
PALETTE    = 12
HUE_IDX = 13
```

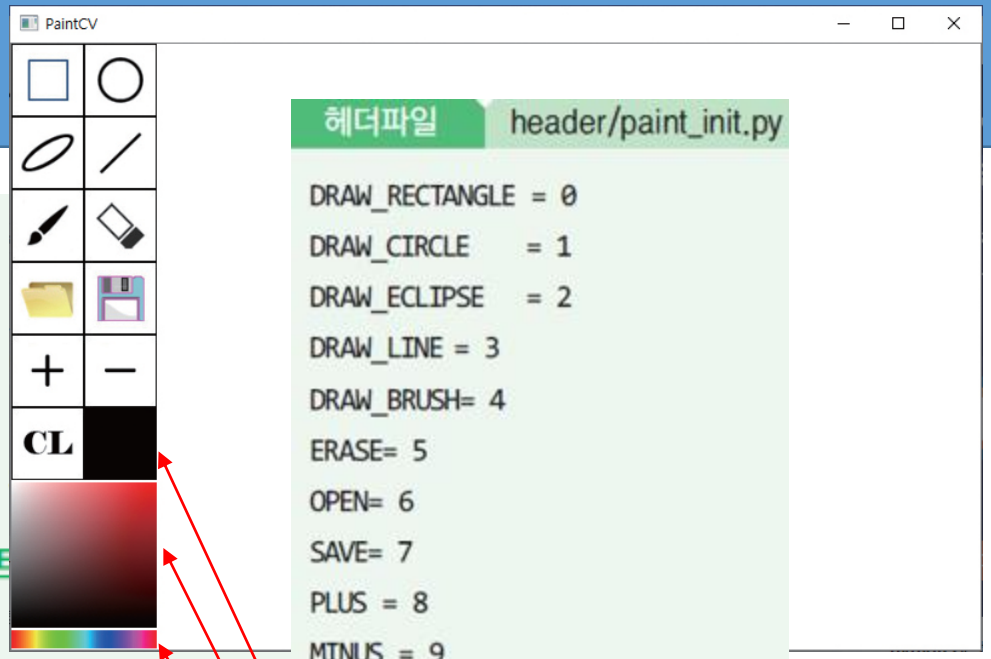
명령 함수 구현

```
01  def command(mode):
02      global icons, image, Color, hue
03
04      if mode == PALETTE:                          # 색상
05          pixel = image[pt2[::-1]]                 # 팔레트
06          x, y, w, h = icons[COLOR]
07          image[y:y+h-1, x:x+w-1] = pixel          # 색상 아이콘 영역에 pi
08          Color = tuple(map(int, pixel))           # 정수형 튜플 변환
09
10      elif mode == HUE_IDX:                        # 색상 인덱스 클릭 시
11          create_colorPlatte(image, pt2[0], icons[PALETTE])  # x 좌표로 팔레트 다시 그림
12
13      cv2.imshow("PaintCV", image)
```

클릭좌표는 (x, y)를 영상좌표 (y, x)로 변환

테두리 제외

색상 아이콘 사각형

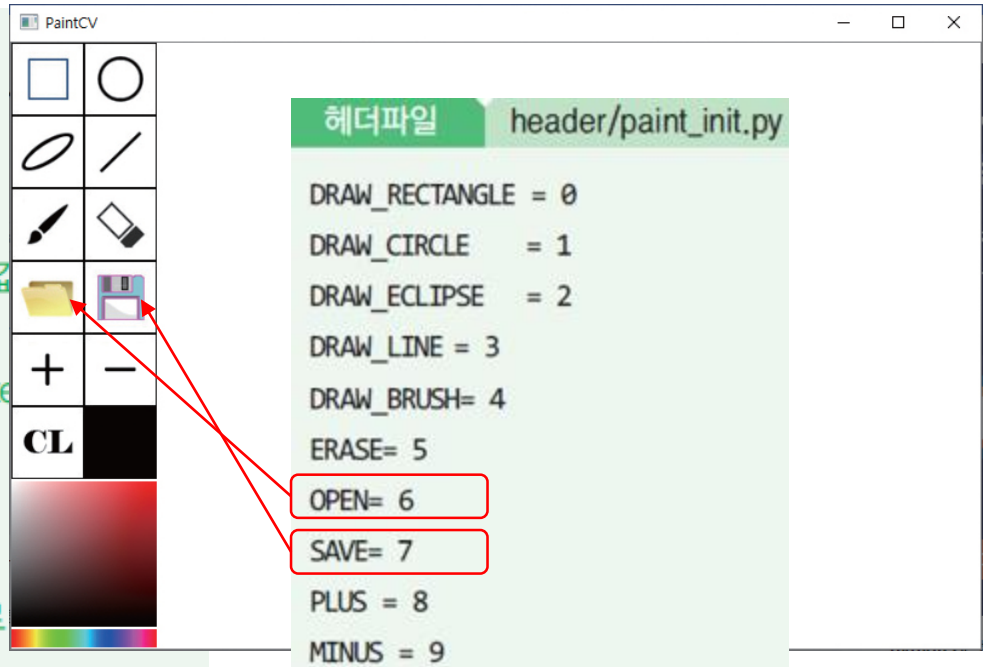


헤더파일 header/paint_init.py

```
DRAW_RECTANGLE = 0
DRAW_CIRCLE    = 1
DRAW_ECLIPSE   = 2
DRAW_LINE = 3
DRAW_BRUSH= 4
ERASE= 5
OPEN= 6
SAVE= 7
PLUS = 8
MINUS = 9
CREAR = 10
COLOR = 11
PALETTE    = 12
HUE_IDX = 13
```

```python
def command(mode):
    global icons, image, canvas, Color, hue, mouse_mode

    if mode == PALETTE:                                  # 색상 팔레트 클릭 시
        pixel = image[pt2[::-1]]                         # 팔레트 클릭 좌표 화소값
        x, y, w, h = icons[COLOR]
        image[y:y+h-1, x:x+w-1] = pixel                  # 색상 아이콘 영역에 pixe
        Color = tuple(map(int, pixel))

    elif mode == HUE_IDX:                                # 색상 인덱스 클릭 시
        create_colorPlatte(image, pt2[0], icons[PALETTE])        # 팔레트 새로

    elif mode == OPEN:                                   # 영상파일 열기
        tmp = cv2.imread("images/my_picture.jpg", cv2.IMREAD_COLOR)
        cv2.resize(tmp, canvas.shape[1::-1], canvas)

    elif mode == SAVE:
        cv2.imwrite("images/my_save.jpg", canvas)
```

캔바스 shape (row, col, depth)를 size(width, height)로 전환



헤더파일 header/paint_init.py

```python
DRAW_RECTANGLE = 0
DRAW_CIRCLE    = 1
DRAW_ECLIPSE   = 2
DRAW_LINE = 3
DRAW_BRUSH= 4
ERASE= 5
OPEN= 6
SAVE= 7
PLUS = 8
MINUS = 9
CREAR = 10
COLOR = 11
PALETTE    = 12
HUE_IDX = 13
```

```
20        elif mode == PLUS:                          # 캔버스 영상 밝
21              val = np.full(canvas.shape, 10, np.uint8)`   # 증가 화소값 행
22              cv2.add(canvas, val, canvas)
23
24        elif mode == MINUS:                         # 캔버스 영상 어
25              val = np.full(canvas.shape, 10, np.uint8)    # 증가 화소값 행
26              cv2.subtract(canvas, val, canvas)
27
28        elif mode == CREAR:                         # 캔버스 영역 전체 지우기
29              canvas[:] = (255, 255, 255)           # 캔버스를 흰색으로
30              mouse_mode = 0                        # 마우스 상태 초기화
31
32  cv2.imshow("PaintCV", image)
```

헤더파일 header/paint_init.py

```
DRAW_RECTANGLE = 0
DRAW_CIRCLE    = 1
DRAW_ECLIPSE   = 2
DRAW_LINE = 3
DRAW_BRUSH= 4
ERASE= 5
OPEN= 6
SAVE= 7
PLUS = 8
MINUS = 9
CREAR = 10
COLOR = 11
PALETTE     = 12
HUE_IDX = 13
```

2. 얼굴 검출 및 성별 분류

❖얼굴 검출 및 인식 응용

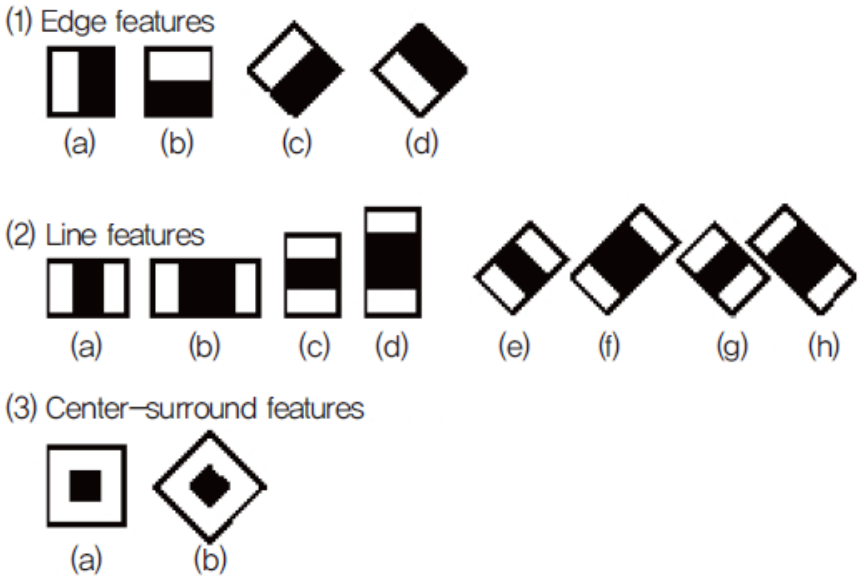
- 디지털 카메라에서도 얼굴 검출 후 액정에 표시
- 페이스북이나 구글에 사진을 올리면 얼굴 영역 검출

〈그림 11.3.1〉 얼굴 검출 응용 예

하르 기반 분류기

❖ 논문 “Rapid Object Detection Using a Boosted Cascade of Simple Features”

- 얼굴과 얼굴이 아닌 것의 차이를 효율적으로 보여줄 수 있는 하르 유사 특징(Haar-like features)을 이용한 방법 제안
- 특징값 정의
  - 흰색 영역의 화소값 합
  - 검은색 영역의 화소값 합
  - 두 영역의 차



〈그림 11.3.2〉 하르 유사 특징

하르 기반 분류기

https://www.youtube.com/watch?v=hPCTwxF0qf4

❖하르 기반 캐스케이드 분류기

- 여러 개의 검출기를 순차적으로 사용
- 처음에 간단한 검출기를 적용하고, 진행할수록 복잡한 검출기 적용
- 단순 검출기를 통과한 후보에만 시간이 많이 걸리는 강력한 검출기가 적용 → 검출 속도 크게 향상

❖OpenCV의 캐스케이드 분류기

- 1,000개 이상의 얼굴 영상과 10,000개 이상의 얼굴이 아닌 영상을 사용하여 학습

하르 기반 분류기

❖분류기 파일 https://github.com/opencv/opencv/tree/master/data/haarcascades

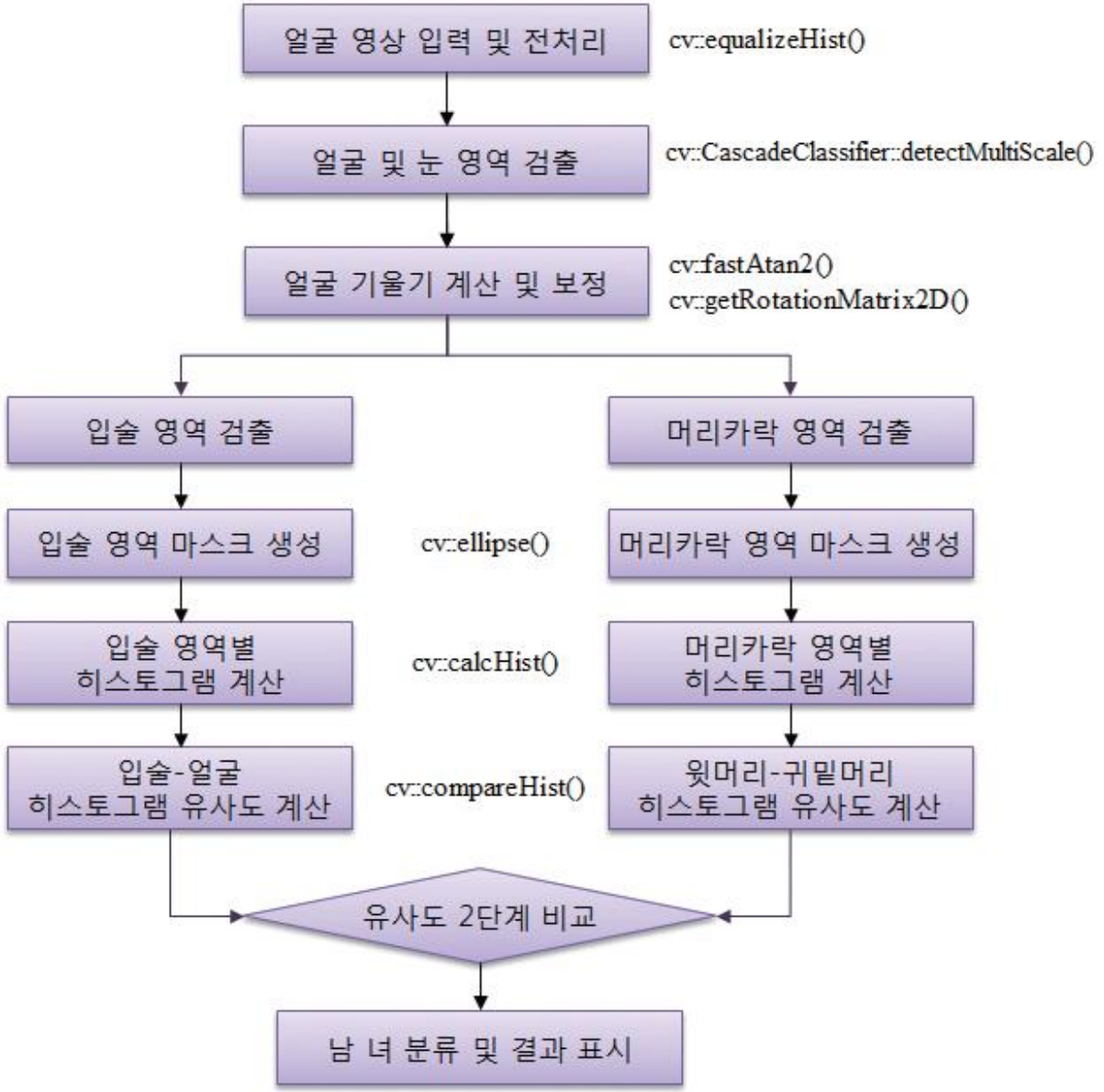
- opencv/sources/data/haarcascades 폴더

〈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

얼굴 검출 및 성별 분류

❖전체 프로그램 구성



얼굴 검출

예제 11.2.1 얼굴 검출 – 06.detect_face.py

```
01  import cv2, numpy as np
02
03  def preprocessing(no):                                  # 전처리 수행 함수
04      image = cv2.imread('images/face/%2d.jpg' %no, cv2.IMREAD_COLOR)
05      if image is None: return None, None
06      gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)      # 명암도 영상 변환
07      gray = cv2.equalizeHist(gray)                       # 히스토그램 평활화
08      return image, gray
09
10  face_cascade = cv2.CascadeClassifier("haarcascade_frontalface_alt2.xml") # 정면 얼굴 검출기
11  eye_cascade = cv2.CascadeClassifier("haarcascade_eye.xml")                # 눈 검출기
12  image, gray = preprocessing(34)                                           # 전처리
13  if image is None: raise Exception("영상파일 읽기 에러")
14
```

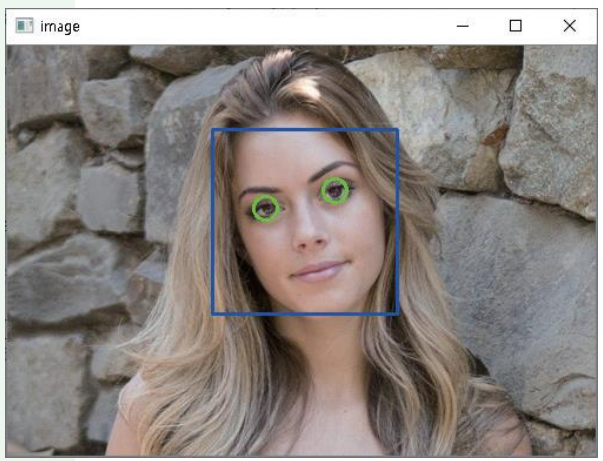
영상 확대 비율

이웃 후보 개수

최소 얼굴 크기

```
15  faces = face_cascade.detectMultiScale(gray, 1.1, 2, 0, (100, 100))          # 얼굴 검출
16  if faces.any():                                                             # 얼굴 사각형 검출되면
17      x, y, w, h = faces[0]
18      face_image = image[y:y+h, x:x+w]                                        # 얼굴 영역 영상 가져오기
19      eyes = eye_cascade.detectMultiScale(face_image, 1.15, 7, 0, (25, 20))   # 눈 검출 수행
20      if len(eyes) == 2:                                                      # 눈 사각형이 검출되면
21          for ex, ey, ew, eh in eyes:
22              center = (x + ex + ew//2, y + ey + eh//2)                       # 중심점 계산
23              cv2.circle(image, center, 10, (0, 255, 0), 2)                   # 눈 중심에 원 그리기
24      else:
25          print("눈 미검출")
26
27      cv2.rectangle(image, faces[0], (255, 0, 0), 2)                          # 얼굴 검출 사각형 그리기
28      cv2.imshow("image", image)
29  else:
30      print("얼굴 미검출")
31  cv2.waitKey()
```

각 눈의 중심 좌표



얼굴 검출

❖CascadeClassifier::detectMultiScale() 함수 사용

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

성별 분류

❖여자와 남자의 구분에 대해서 두 가지를 가정

1) 입술의 색깔이 남자에 비해 여자가 더 붉다.

2) 머리카락의 길이가 남자에 비해 여자가 더 길다.

* 이 가정이 항상 옳은 것은 아님
- 여성이라도 머리카락이 짧을 수 있음 / 남자라도 머리카락이 길 수 있음
- 남자가 입술 화장을 할 수도 있으며, 입술 화장을 하지 않는 여자도 있음

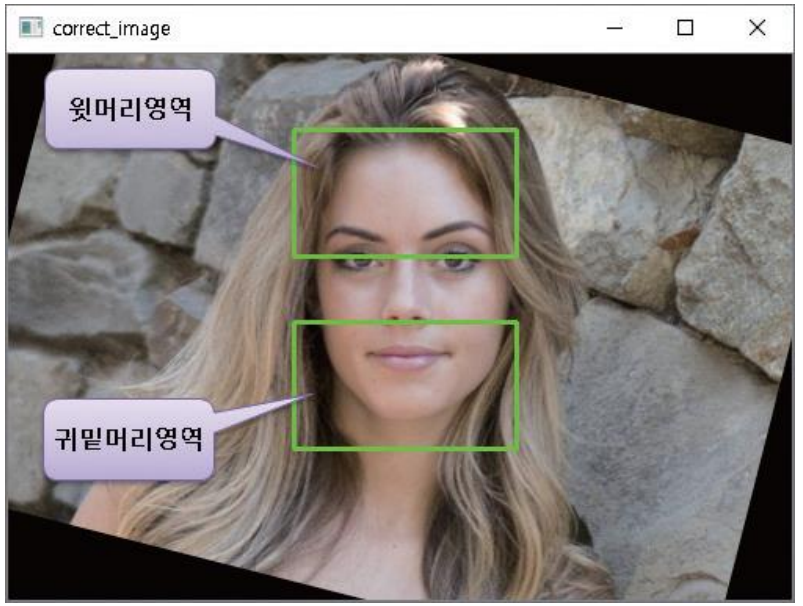
▪ 두 가지 가정을 전제로

• 간단하면서도 쉽게 남녀를 구분 할 수 있다.

성별 분류

- 여자의 입술 색상이 더 붉다는 가정
  - 입술 영역의 색상과 얼굴 영역의 색상을 비교
    → 두 영역의 색상이 비슷하면 남자로 분류하고, 다르면 여자로 분류

- 여자의 긴 머리카락과 남자의 짧은 머리카락을 어떻게 인식할까?
  - 귀밑 영역에 머리카락이 있는지를 검사하는 것

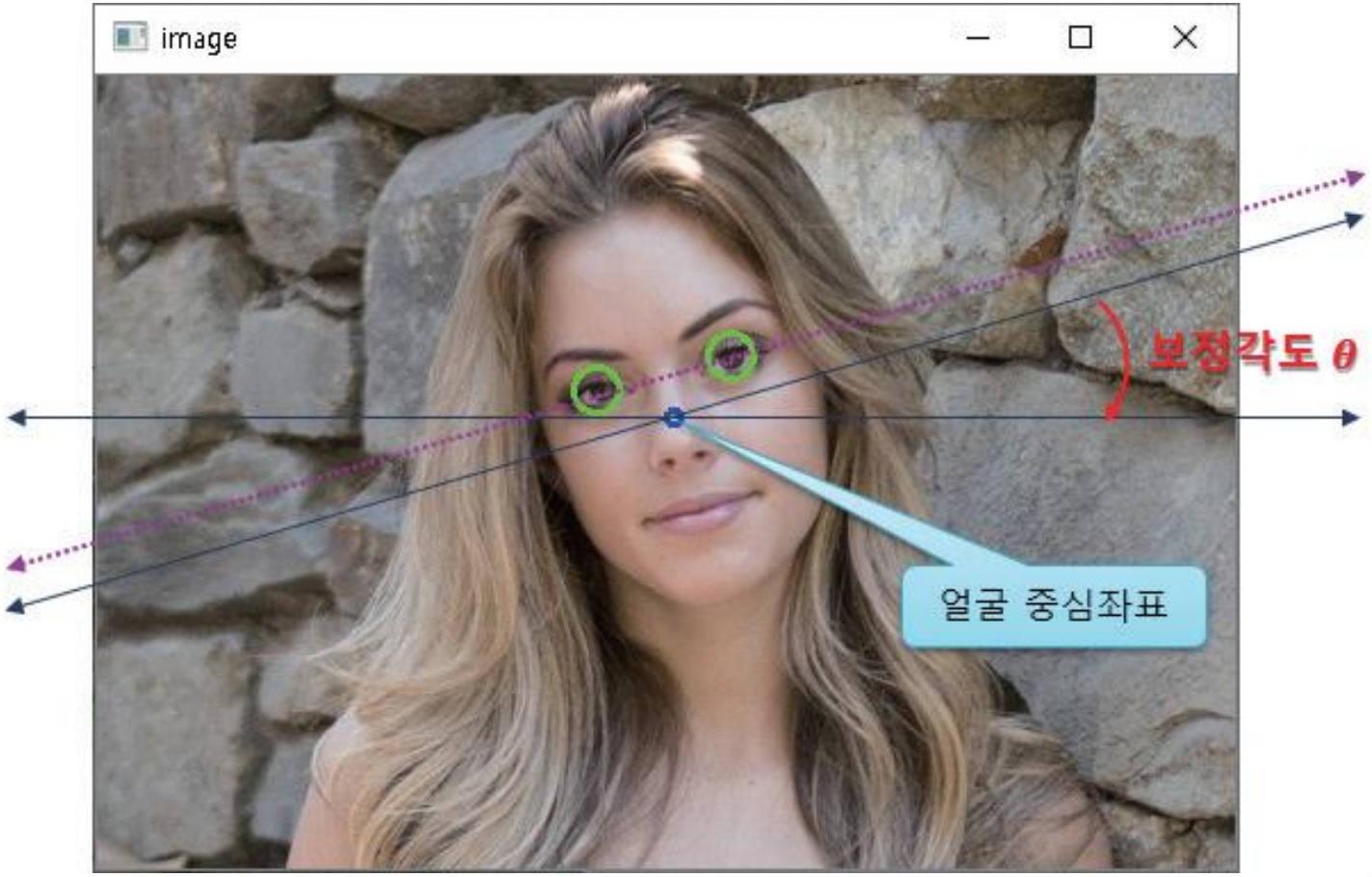


- cv::compareHist() 함수로 유사도 비교

얼굴의 기울기 보정

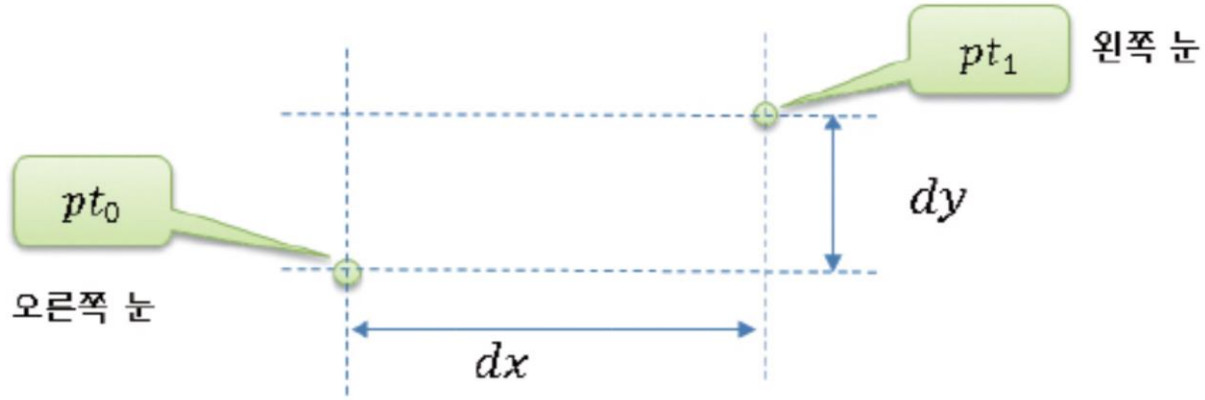
❖얼굴의 기울어진 각도는 두 눈의 중심좌표 이용

- 눈 검출 필요 , 중심점에서 기울기만큼 회전



얼굴의 기울기 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

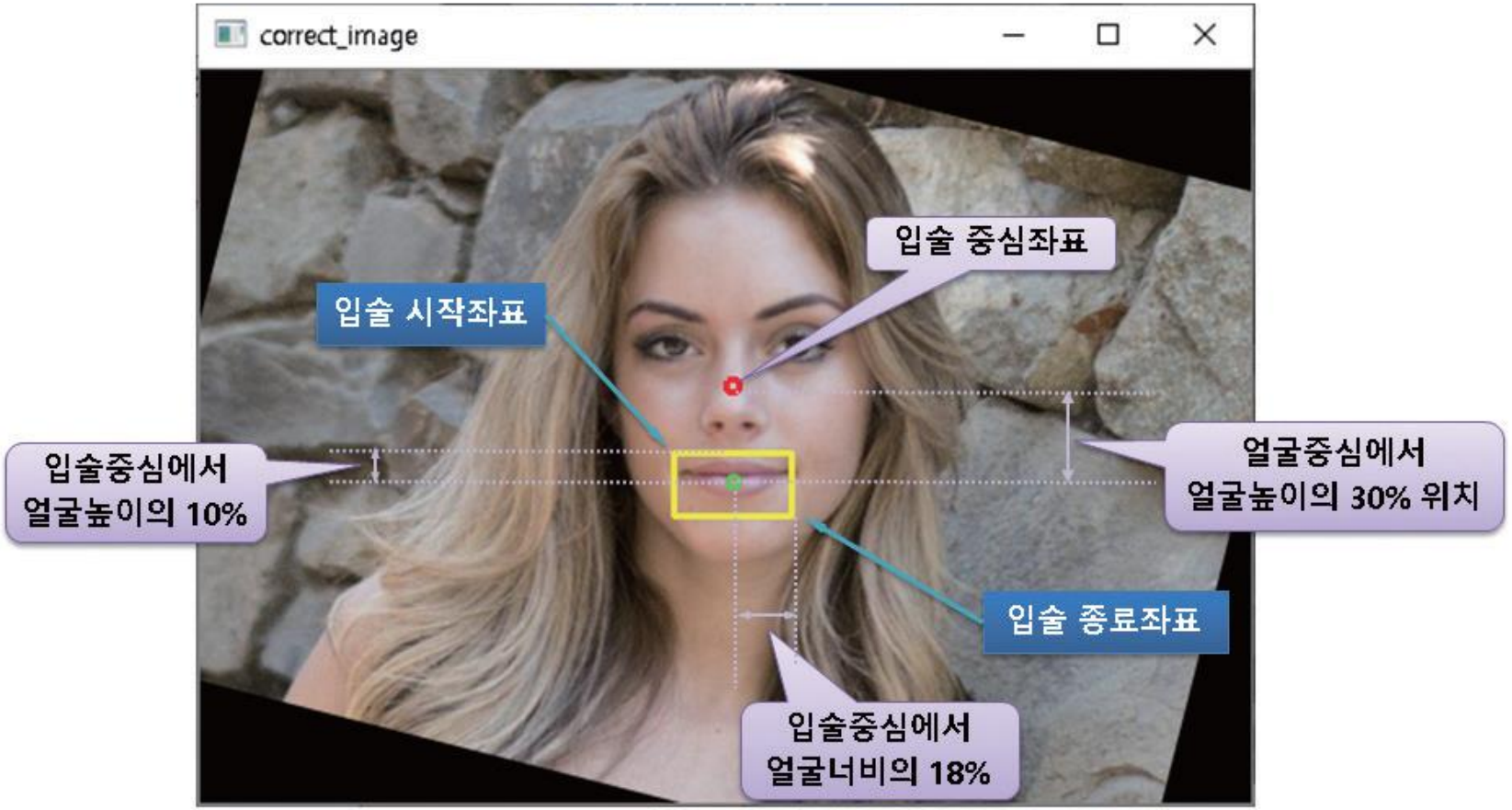
얼굴의 기울기 보정

header/utils.py

```
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
```

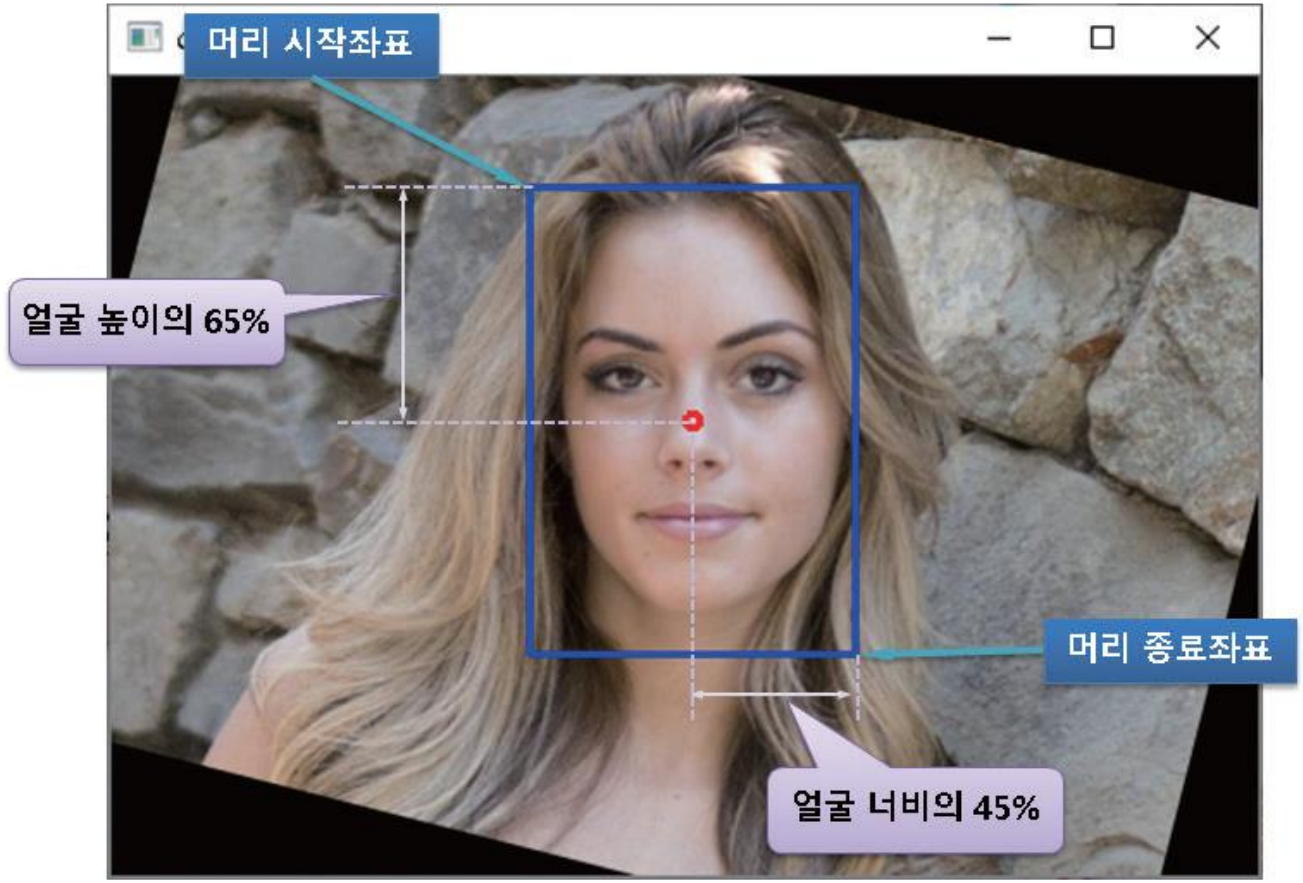
입술 영역 및 머리 영역 검출

- 입술 영역 계산



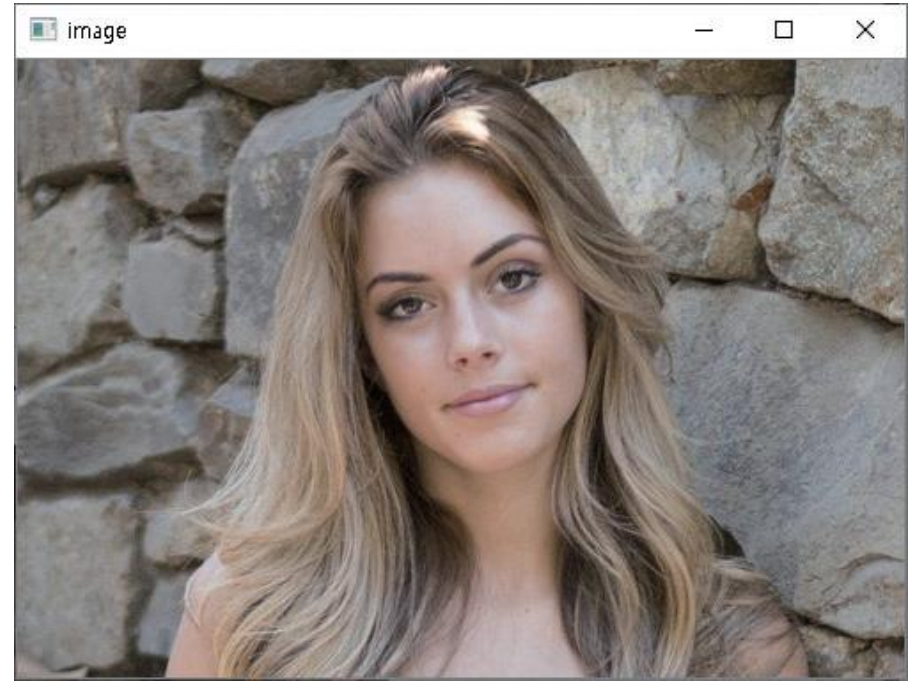
입술 영역 및 머리 영역 검출

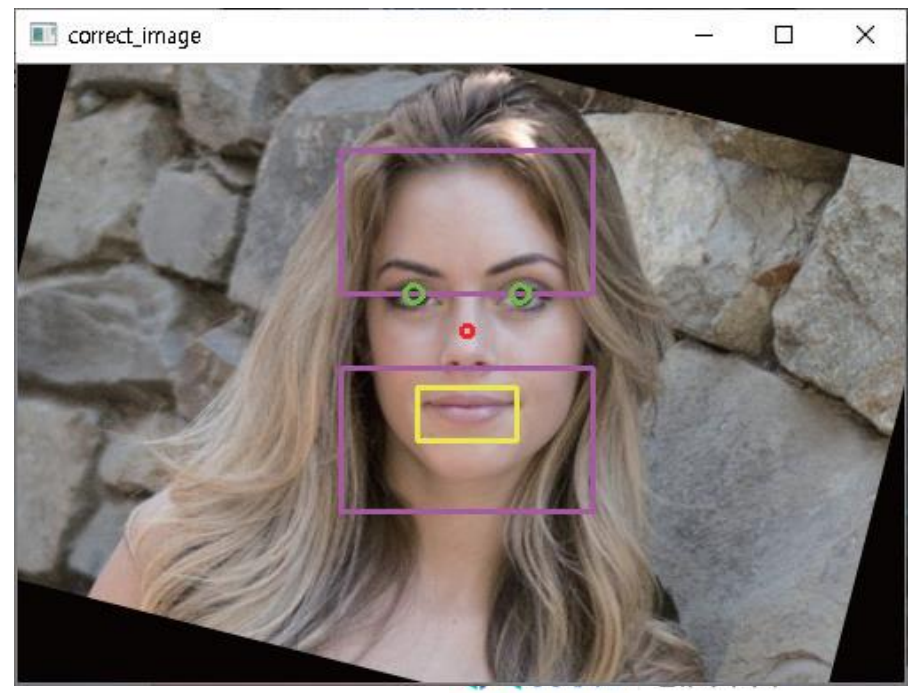
- 윗머리 및 귀밑머리 영역 계산



```python
def define_roi(pt, size):
    return np.ravel((pt, size)).astype('int')          # 2 원소 튜플 2개 → 4 원소 튜플 1개

def detect_object(center, face):
    w, h = np.array(face[2:4])                         # 얼굴 영역 크기(w, h)
    center = np.array(center)                          # 얼굴 중심좌표 ndarray 객체로 변경
    gap1 = np.multiply((w, h) * (0.45, 0.65))          # 얼굴 영역 비율 크기 45%, 65%
    gap2 = np.multiply((w, h) * (0.18, 0.1))           # 입술 영역 비율 크기18%, 10%

    pt1 = center - gap1                                # 좌상단 평행이동– 머리 시작 좌표
    pt2 = center + gap1                                # 우하단 평행이동– 머리 종료 좌표
    hair = define_roi(pt1, pt2 - pt1)                  # 전체 머리 영역

    size = np.multiply(hair[2:4], (1, 0.4))            # 머리카락 영역 높이 40%
    hair1 = define_roi(pt1, size)                      # 윗머리 영역 (x, y, w, h)
    hair2 = define_roi(pt2-size, size)                 # 귀밑머리 영역

    lip_center = center + (0, int(h*0.3))              # 입술 영역 중심 좌표 – 30%
    lip1 = lip_center - gap2                           # 좌상단 평행이동
    lip2 = lip_center + gap2                           # 우하단 평행이동
    lip = define_roi(lip1, lip2 - lip1)                # 입술 영역

    return [hair1, hair2, lip, hair]                   # 각 영역을 리스트 구성 후 반환
```

```python
if faces.any() :
    x, y, w, h = faces[0]
    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

    if len(eyes) == 2:
        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) # 회전 보정
        rois = detect_object(face_center, faces[0])          # 머리카락/입술 영역 계산

        cv2.rectangle(corr_image, rois[0], (255, 0, 255), 2)    # 윗머리 영역
        cv2.rectangle(corr_image, rois[1], (255, 0, 255), 2)    # 귀밑머리 영역
        cv2.rectangle(corr_image, rois[2], (255, 0, 0), 2)      # 입술 영역

        cv2.circle(corr_image, tuple(corr_center[0]), 5, (0, 255, 0), 2)    # 보정 눈 좌표
        cv2.circle(corr_image, tuple(corr_center[1]), 5, (0, 255, 0), 2)    # 보정 눈 좌표
        cv2.circle(corr_image, face_center, 3, (0, 0, 255), 2) # 얼굴 중심좌표
        cv2.imshow("correct_image", corr_image)
    else:
        print("눈 미검출")
else:
    print("얼굴 미검출")
cv2.imshow("image", image)
cv2.waitKey(0)
```



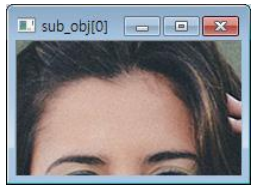


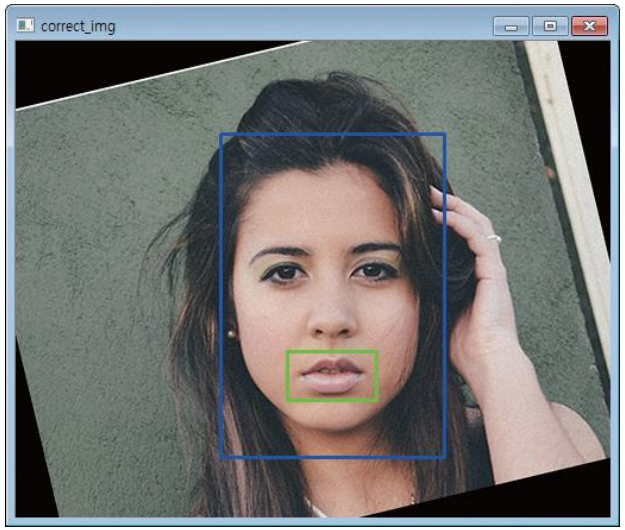
히스토그램 비교

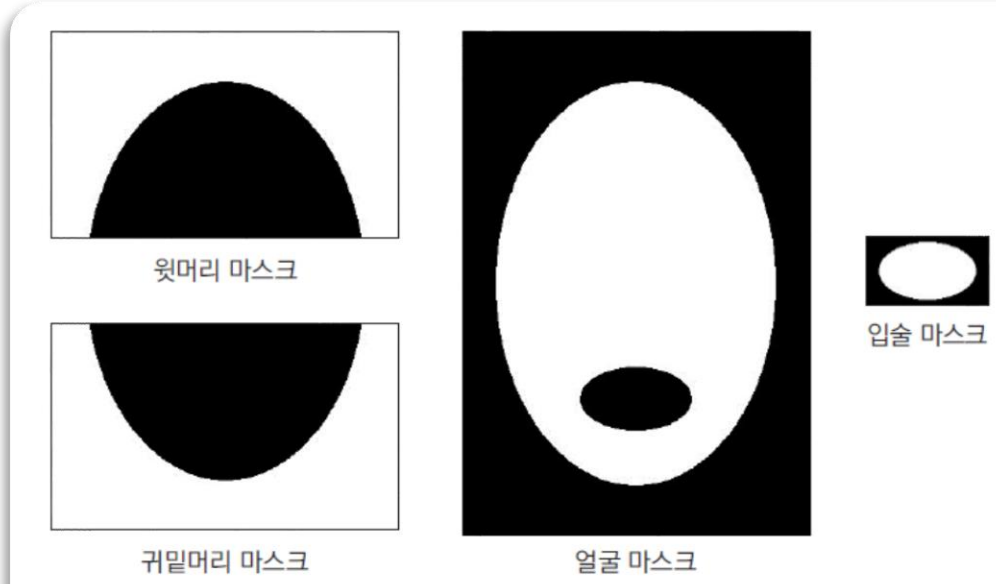
❖히스토그램 산출과정에서 각 영상에 해당하는 부분만 계산

→ 마스크 이용

- 입술 영역 히스토그램은 입술 영역만
- 얼굴 영역 히스토그램은 얼굴 영역만 계산
- 윗머리/ 귀밑머리도 그 영역만 계산







〈그림 11.2.9〉 얼굴 영상 히스토그램을 위한 마스크 영상의 예

- 마스크를 이용하여 각 서브 영역의 히스토그램을 생성

  Header/haar_histogram.py

```
def calc_histo(image, rois, masks):
    bsize = (64, 64, 64)                                    # 히스토그램 계급 개수
    ranges = (0,256, 0,256, 0,256)                          # 각 채널 빈도 범위
    subs = [image[y:y+h, x:x+w] for x, y, w, h in rois]     # 관심 영역 참조로 영상 생성

    hists = [cv2.calcHist([sub], [0, 1, 2], mask, bsize, ranges, 3)
                    for sub, mask in zip(subs, masks)]      # 관심 영역 영상 히스토그램
    hists = [h / np.sum(h) for h in hists]                  # 히스토그램값 정규화
    sim1= cv2.compareHist(hists[2], hists[3], cv2.HISTCMP_CORREL)   # 입술-얼굴 유사도
    sim2= cv2.compareHist(hists[0], hists[1], cv2.HISTCMP_CORREL)   # 윗-귀밑머리 유사도
    return sim1, sim2
```

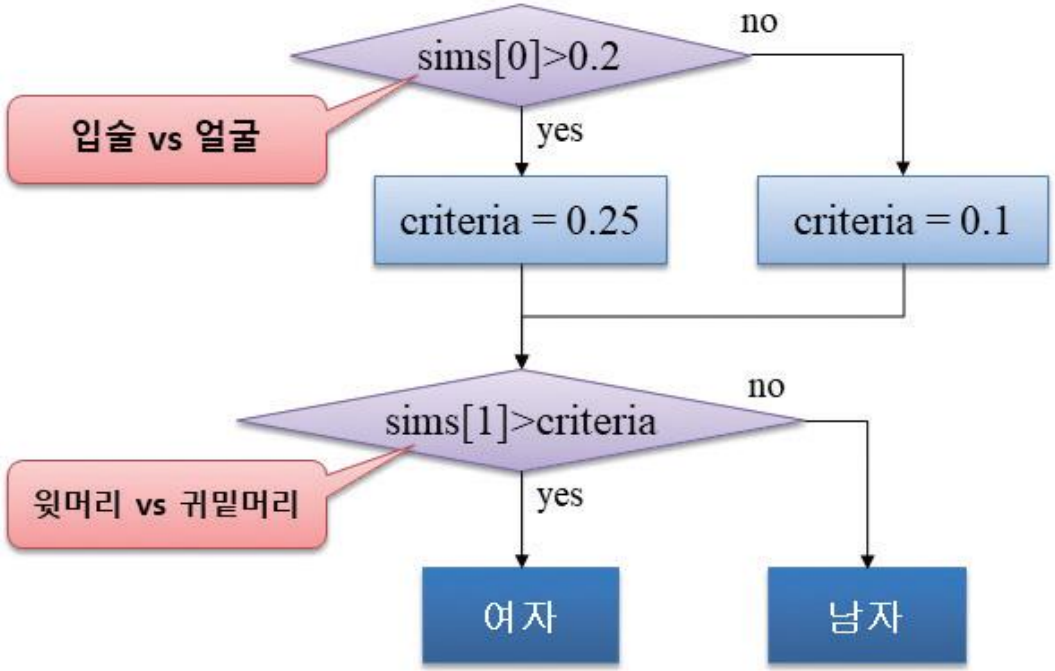
성별 분류

❖윗머리와 귀밑머리의 유사도 (criteria1)

- 크다 → 두 영역 색상 비슷
  → 귀밑에 머리카락 있을 확률↑
  → 여자 확률↑

❖얼굴과 입술의 유사도 (criteria2)

- 작다 → 입술과 얼굴 색상 다름
  → 입술색이 도드라짐
  → 여자 확률↑



화살표키로 다음 영상 분류 수행