8장 과제

201935282 송우석

1. 순방향 사상과 역방향 사상에 대해 설명하고 장단점을 비교하시오.

순방향 사상은 입력 영상의 좌표를 중심으로 목적 영상의 좌표를 계산하여 화소의 위치를 변환하는 방식이다. 일반적으로 입력영상과 목적영상이 크기가 같을 때 유용하게 사용된다. 두 영상의 크기가 달라지면, 홀이나 오버랩의 문제가 발생한다.

역방향 사상은 목적 영상의 좌표를 주심으로 역변환을 계산하여 해당하는 원본 영상의 좌표를 찾아서 화소값을 가져오는 방식이다.

2. 홀과 오버랩에 대하여 설명하시오.

홀은 입력 영상의 좌표들로 목적 영상의 좌표를 만드는 과정에서 사상되지 않은 화소를 가리킨다. 보통 영상을 확대, 회전할 때 발생한다.
오버랩은 영상을 축소할 때 주로 발생한다. 이것은 입력 영상의 여러 화소들이 목적 영상의 한 화소로 사상 되는 것을 말한다.

3. 보간법이 필요한 이유와 OpenCV에서 보간 방법을 가리키는 옵션 상수를 설명하시오.

홀의 화소와 오버랩 되지 않게 화소들을 배치하여 목적 영상을 만드는 기법을 보간법이라고 한다.

OpenCV에서 보간 방법을 가리키는 옵션상수는 cv2.INTER_NEAREST, cv2.INTER_LINEAR, cv2.INTER_CUBIC, cv2.INTER_AREA, cv2.INTER_LANCZOS4가 있다.

4. 최근접 이웃 보간법에 대해 기술하시오.

최근접 이웃 보간법은 목적 영상을 만드는 과정에서 홀이 되어 화소값을 할당 받지 못한 위치에 값을 찾을 때, 그 위치에 가장 가깝게 이웃한 입력 영상의 화소값을 가져오는 방법이다.

5. 양선형 보간법의 과정을 상세히 설명하시오.

두 개 화소 값을 알고 있을 때 그 값으로 직선을 그리면 직선 위에 위치한 화소들의 값을 구할 수 있다. 양선형 보간법은 이와 같은 선형 보간을 두 번에 걸쳐 수행한다.

6. 중심점을 기준으로 회전 변환을 수행하려면 어떠한 과정을 거쳐야하는지 변환 행렬의 곱으로 나타내고 설명하시오.

회전 변환은 원점을 중심으로 회전하므로 회전 기준점(중심점)을 원점으로 평행이동을 한다. 그 뒤 회전 변환을 수행하고 다시 중심점의 원래 좌표로 평행이동을 한다.

$$T = \begin{bmatrix} 1 & 0 & -c_x \\ 0 & 1 & -c_y \\ 0 & 0 & 1 \end{bmatrix}$$

$$R = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

M = R * T 이고 변환행렬 M을 다시 중심점으로 옮긴다.

7. 어파인 변환을 수행하는 OpenCV 함수들을 예시하고 인수들에 대해 설명하시오.

cv2.warpAffine(src, M, dsize [, dst [, flags[, borderMode [, borderValue]]]]) → dst
src - 입력 영상, dst - 반환 영상, M - 어파인 변환 행렬, dsize - 반환 영상의 크기, flags - 보간 방법

cv2.getAffineTransform(src, dst) → retval
src - 입력 영상 좌표 3개, dst - 목적 영상 좌표 3개

cv2.getRotationMatrix2D(center, angle, scale) → retval
center - 회전의 중심점, angle - 회전 각도, scale - 변경할 크기

8. 원근 변환에 대해서 설명하시오.

원근법은 눈에 보이는 3차원의 세계를 2차원의 평면으로 옮길 때 관찰자가 보는 것 그대로 사물과의 거리를 반영하여 그리는 방법을 말한다. 그리고 이 원근법을 영상 좌표계에서 표현하는 것이 원근 투시 변환이다. 원근 변환에서는 주로 동차 좌표계를 사용하는 것이 편리하다.

9. 원근 변환을 수행하는 OpenCV 함수들을 예시하고 각 인수들을 설명하시오.

cv2.getPerspectiveTransform(src, dst [, solveMethod]) → retval
src - 입력 영상 4개 좌표, dst - 목적 영상 4개 좌표, borderMode - 경계 지정 방법

cv2.warpPerspective(src, M, dsize[, dst[, flags[, borderMode[, borderValue]]]]) →dst
src - 입력 영상, M - 원근 변환 행렬 , dsize - 결과 영상 크기, dst - 결과 영상, flags - 보간방법

cv2.transform(Src, M) → dst
src - 입력 좌표 행렬, dst -결과 좌표 행렬, M - 원근 변환 행렬

10번 문제

```
import numpy as np, cv2


def contain(p,shape):
    return 0 <= p[0] < shape[0] and 0 <= p[1] <shape[1]

def translate(img, pt):
    dst = np.zeros(img.shape, img.dtype)            # 목적 영상 생성
    for i in range(img.shape[0]):                           # 목적 영상 순회 - 역방향 사상
        for j in range(img.shape[1]):
            x, y = np.subtract((j, i) , pt)
            if contain((y, x), img.shape):
                dst[i, j] = img[y, x]
    return dst

image = cv2.imread("translate.jpg", cv2.IMREAD_GRAYSCALE)
if image is None: raise Exception("영상 파일 읽기 에러")

dst1 = translate(image, (50,60))

M = np.float32([[1, 0, 50], [0, 1, 60]])
# 평행이동하는 opencv함수 warpAffine
dst2 = cv2.warpAffine(image, M, (image.shape[1], image.shape[0]))


cv2.imshow("image", image)
cv2.imshow("dst1", dst1)
cv2.imshow("dst2", dst2)
cv2.waitKey(0)
```

11번 문제

```
import numpy as np, cv2
from Common.interpolation import bilinear_value
from Common.utils import contain   , ck_time               # 사각형으로 범위 확인 함수

def rotate(img, degree, pt):
    dst = np.zeros(img.shape[:2], img.dtype)                    # 목적 영상 생성
    radian = (degree/180) * np.pi                              # 회전 각도 - 라디언
    sin, cos = np.sin(radian), np.cos(radian)   # 사인, 코사인 값 미리 계산

    for i in range(img.shape[0]):                                 # 목적 영상 순회 - 역방향 사상
        for j in range(img.shape[1]):
            jj, ii = np.subtract((j, i), pt)                # 중심좌표 평행이동,
            y = -jj * sin + ii * cos               # 회선 변환 수식
            x =  jj * cos + ii * sin
            x, y = np.add((x, y), pt)
            if contain((y, x), img.shape):                          # 입력 영상의 범위 확인
                dst[i, j] = bilinear_value(img, [x, y])           # 화소값 양선형 보간
    return dst

image = cv2.imread("translate.jpg", cv2.IMREAD_GRAYSCALE)
if image is None: raise Exception("영상 파일 읽기 에러")

center = (100, 100)
```

```
dst1 = rotate(image,30, center)

cv2.imshow("dst1", dst1)
cv2.waitKey(0)
```

12번 문제

```
import numpy as np,  cv2
from Common.interpolation import bilinear_value
from Common.utils import contain   , ck_time                # 사각형으로 범위 확인 함수

def rotate(img, degree):
    dst = np.zeros(img.shape[:2], img.dtype)                    # 목적 영상 생성
    radian = (degree/180) * np.pi                               # 회전 각도 - 라디언
    sin, cos = np.sin(radian), np.cos(radian)   # 사인, 코사인 값 미리 계산

    for i in range(img.shape[0]):                                       # 목적 영상 순회 - 역방향 사상
        for j in range(img.shape[1]):
            y = -j * sin + i * cos
            x =  j * cos + i * sin                  # 회선 변환 수식
            if contain((y, x), img.shape):              # 입력 영상의 범위 확인
                dst[i, j] = bilinear_value(img, [x, y])           # 화소값 양선형 보간
    return dst

image = cv2.imread('affine.jpg', cv2.IMREAD_COLOR)
if image is None: raise Exception("영상 파일을 읽기 에러")

b,g,r = cv2.split(image)
degree = 30
# 원점을 중심으로 회전 후 merge
b = rotate(b,30)
g = rotate(g,30)
r = rotate(r,30)

dst = cv2.merge([b,g,r])

cv2.imshow("dst", dst)
cv2.waitKey(0)
```

13번 문제

```
import numpy as np, cv2
from Common.interpolation import rotate_pt

def calc_d(pts):
    d1 = np.subtract(pts[1], pts[0]).astype(float)        # 두 좌표간 차분 계산
    return d1  # 두 각도 간의 차분

def draw_point(x, y, color):
    pts.append([x,y])
    print("좌표:", len(pts), [x,y])
    cv2.circle(tmp, (x, y), 2, color, 2)  # 중심 좌표 표시
    cv2.imshow("image", tmp)

def onMouse(event, x, y, flags, param):
    global tmp, pts
    if (event == cv2.EVENT_LBUTTONDOWN and len(pts) == 0):  draw_point(x, y, (0,0,255))
    if (event == cv2.EVENT_RBUTTONDOWN and len(pts) == 1):
        draw_point(x, y, (0,255, 0))
```

```
    if len(pts) == 2:
        dxy = calc_d(pts)
        print("차분x : %3.2f" % dxy[0])
        print("차분y : %3.2f" % dxy[1])
        cv2.line(image, pts[0], pts[1], (0,255,0),1)
        pts = []

image = cv2.imread('affine.jpg', cv2.IMREAD_GRAYSCALE)
if image is None: raise Exception("영상 파일을 읽기 에러")
tmp = np.copy(image)
pts = []

cv2.imshow("image", image)
cv2.setMouseCallback("image", onMouse, 0)
cv2.waitKey(0)
```

14번 문제

```
import numpy as np, cv2
from Common.interpolation import rotate_pt

def calc_d(pts):
    d1 = np.subtract(pts[1], pts[0]).astype(float)        # 두 좌표간 차분 계산

    return d1  #차분

def draw_point(x, y, color):
    pts.append([x,y])
    print("좌표:", len(pts), [x,y])
    cv2.circle(tmp, (x, y), 2, color, 2)  # 중심 좌표 표시
    cv2.imshow("image", tmp)

def onMouse(event, x, y, flags, param):
    global tmp, pts
    if (event == cv2.EVENT_LBUTTONDOWN and len(pts) == 0):  draw_point(x, y, (0,0,255))
    if (event == cv2.EVENT_RBUTTONDOWN and len(pts) == 1):
        draw_point(x, y, (0,255, 0))

    if len(pts) == 2:
        dxy = calc_d(pts)  # 차분 계산
        print("차분x : %3.2f" % dxy[0])
        print("차분y : %3.2f" % dxy[1])

        M = np.float32([[1, 0, dxy[0]], [0, 1, dxy[1]]])
        dst = cv2.warpAffine(image, M, (image.shape[1], image.shape[0]))
        cv2.imshow("image", dst)
        tmp = np.copy(image)                    # 임시 행렬 초기화
        pts = []

image = cv2.imread('affine.jpg', cv2.IMREAD_GRAYSCALE)
if image is None: raise Exception("영상 파일을 읽기 에러")
tmp = np.copy(image)
pts = []

cv2.imshow("image", image)
cv2.setMouseCallback("image", onMouse, 0)
cv2.waitKey(0)
```

15번 문제

```
import numpy as np, cv2
from Common.interpolation import rotate_pt

def calc_d(pts):
    d1 = np.subtract(pts[1], pts[0]).astype(float)        # 두 좌표간 차분 계산
    angle1 = cv2.fastAtan2(d1[1], d1[0])  # 차분으로 각도 계산

    return angle1  # 두 각도 간의 차분

def draw_point(x, y, color):
    pts.append([x,y])
    print("좌표:", len(pts), [x,y])
    cv2.circle(tmp, (x, y), 2, color, 2)  # 중심 좌표 표시
    cv2.imshow("image", tmp)

def onMouse(event, x, y, flags, param):
    global tmp, pts
    if (event == cv2.EVENT_LBUTTONDOWN and len(pts) == 0):  draw_point(x, y, (0,0,255))
    if (event == cv2.EVENT_RBUTTONDOWN and len(pts) == 1):
        draw_point(x, y, (0,255, 0))

    if len(pts) == 2:
        angle = calc_d(pts)  # 회전각 계산
        dst = rotate_pt(image, angle, pts[0])  # 사용자 정의 함수 회전 수행

        # 평행이동하는 opencv함수 warpAffine
        cv2.imshow("image", dst)

        tmp = np.copy(image)                    # 임시 행렬 초기화
        pts = []

image = cv2.imread('affine.jpg', cv2.IMREAD_GRAYSCALE)
if image is None: raise Exception("영상 파일을 읽기 에러")
tmp = np.copy(image)
pts = []

cv2.imshow("image", image)
cv2.setMouseCallback("image", onMouse, 0)
cv2.waitKey(0)
```

16번 문제

```
import numpy as np, cv2
from Common.utils import contain_pts


def draw_rect(img):
    rois = [(p - small, small * 2) for p in pts1]
    for (x, y), (w, h) in np.int32(rois):
        roi = img[y:y + h, x:x + w]
        val = np.full(roi.shape, 80, np.uint8)
        cv2.add(roi, val, roi)
        cv2.rectangle(img, (x, y, w, h), (0, 255, 0), 1)
    cv2.polylines(img, [pts1.astype(int)], True, (0, 255, 0), 1)
    cv2.imshow("select rect", img)


def warp(img):
    perspect_mat = cv2.getPerspectiveTransform(pts1, pts2)
    dst = cv2.warpPerspective(img, perspect_mat, (400, 350), cv2.INTER_CUBIC)
    cv2.imshow("perspective transform", dst)
```

```
def onMouse(event, x, y, flags, param):
    global check
    if event == cv2.EVENT_LBUTTONDOWN:
        for i, p in enumerate(pts1):
            p1, p2 = p - small, p + small
            if contain_pts((x, y), p1, p2): check = i

    if event == cv2.EVENT_LBUTTONUP: check = -1

    if check >= 0:
        pts1[check] = (x, y)
        draw_rect(np.copy(frame))
        warp(np.copy(frame))

capture = cv2.VideoCapture(0)
if not capture.isOpened() : raise Exception("카메라 연결 안됨")

capture.set(cv2.CAP_PROP_FRAME_WIDTH, 600)
capture.set(cv2.CAP_PROP_FRAME_HEIGHT, 800)

while True:
    ret, frame = capture.read()
    if not ret : break

    small = np.array((12, 12))  # 좌표 사각형 크기
    check = -1  # 선택 좌표 사각형 번호 초기화
    pts1 = np.float32([(100, 100), (300, 100), (300, 300), (100, 300)])
    pts2 = np.float32([(0, 0), (400, 0), (400, 350), (0, 350)])  # 목적 영상 4개 좌표

    draw_rect(np.copy(frame))
    cv2.setMouseCallback("select rect", onMouse, 0)
    if cv2.waitKey(30) >= 0:break

capture.release()
```

17번 문제

```
import numpy as np, cv2

image = cv2.imread('affine.jpg', cv2.IMREAD_GRAYSCALE)
if image is None: raise Exception("영상 파일을 읽기 에러")

h,w = image.shape

flip1 = np.array([[-1, 0, w],[0, 1, 0]], dtype=np.float32)
flip2 = np.array([[1, 0, 0],[0, -1, h]], dtype=np.float32)
flip3 = np.array([[-1, 0, w],[0, -1, h]], dtype=np.float32)

dst1 = cv2.warpAffine(image, flip1, (w,h))
dst2 = cv2.warpAffine(image, flip2, (w,h))
dst3 = cv2.warpAffine(image, flip3, (w,h))

cv2.imshow("image", image)
cv2.imshow("flip1", flip1)
cv2.imshow("flip2", flip2)
cv2.imshow("flip3", flip3)
cv2.waitKey(0)
```

18번 문제

```
import numpy as np, cv2
```

```
def draw_bar(img, pt, w, bars):
    pt = np.array(pt, np.int)
    for bar in bars:
        (x,y), h = pt, w*6
        cv2.rectangle(img, (x,y,w,h), (0,0,0),-1)
        if bar == 0:
            y = (y + w * 3 - w // 4)
            h = (w // 2)
            cv2.rectangle(img, (x,y,w,h), (255,255,255),-1)
        pt += (int(w*1.5),0)
```

19번 문제

```
import numpy as np, cv2


def draw_bar(img, pt, w, bars):
    pt = np.array(pt, np.int8)
    for bar in bars:
        (x, y), h = pt, w * 6
        cv2.rectangle(img, (x, y, w, h), (0, 0, 0), -1)
        if bar == 0:
            y = (y + w * 3 - w // 4)
            h = (w // 2)
            cv2.rectangle(img, (x, y, w, h), (255, 255, 255), -1)
        pt += (int(w * 1.5), 0)


c = 200
r, sr, c2, c4 = c // 2, c // 4, c * 2, c * 4
img = np.full((c4, c4, 3),255, np.uint8)
blue, red = (255, 0, 0), (0, 0, 255)

cv2.ellipse(img, (c2, c2), (r, r), 0, 0, 180, blue, -1)
cv2.ellipse(img, (c2, c2), (r, r), 180, 0, 180, red, -1)
cv2.ellipse(img, (c2 + r - sr, c2), (sr, sr), 180, 0, 180, blue, -1)
cv2.ellipse(img, (c2 - sr, c2), (sr, sr), 0, 0, 180, red, -1)

left = (c2 - c * (18)/24, c2 - sr)
right = (c2 + c * (18)/24, c2 -sr)

draw_bar(img, left, c//12, (1,1,1))
draw_bar(img, right, c//12, (0,0,0))


angle = cv2.fastAtan2(2,3)
# img = cv2.warpAffine(img, cv2.getRotationMatrix2D((c2,c2), -angle*2,1), (c4,c4))


cv2.imshow("image", img)
cv2.waitKey()
```