컴퓨터 그래픽스 과제 #05

학과 : 컴퓨터공학과
학번 : 201802161
이름 : 조은빈

1. 구현 코드

* calc_M_harris

```
def calc_M_harris(IxIx, IxIy, IyIy, fsize = 5):
    assert IxIx.shape == IxIy.shape and IxIx.shape == IyIy.shape
    h, w = IxIx.shape
    M = np.zeros((h, w, 2, 2))
    IxIx_pad = my_padding(IxIx, (fsize, fsize))
    IxIy_pad = my_padding(IxIy, (fsize, fsize))
    IyIy_pad = my_padding(IyIy, (fsize, fsize))

    '''for row in range(h):
        for col in range(w):
            M[row, col, 0, 0] = np.sum(IxIx_pad[row:row+fsize, col:col+fsize])
            M[row, col, 0, 1] = np.sum(IxIy_pad[row:row+fsize, col:col+fsize])
            M[row, col, 1, 0] = M[row, col, 0, 1]
            M[row, col, 1, 1] = np.sum(IyIy_pad[row:row+fsize, col:col+fsize])'''

    for row in range(h):
        for col in range(w):
            for f_row in range(fsize):
                for f_col in range(fsize):
                    ##############################
                    # ToDo
                    # 위의 2중 for문을 참고하여 M 완성
                    ##############################
                    M[row, col, 0, 0] += IxIx_pad[row + f_row, col + f_col]
                    M[row, col, 0, 1] += IxIy_pad[row + f_row, col + f_col]
                    M[row, col, 1, 1] += IyIy_pad[row + f_row, col + f_col]
            M[row, col, 1, 0] = M[row, col, 0, 1]

    return M
```

* harris_detector

```python
def harris_detector(src, k = 0.04, threshold_rate = 0.01, fsize=5):
    harris_img = src.copy()
    h, w, c = src.shape
    gray = cv2.cvtColor(src, cv2.COLOR_BGR2GRAY) / 255.
    # calculate Ix, Iy
    Ix, Iy = calc_derivatives(gray)

    # Square of derivatives
    IxIx = Ix**2
    IyIy = Iy**2
    IxIy = Ix * Iy

    start = time.perf_counter()  # 시간 측정 시작
    M_harris = calc_M_harris(IxIx, IyIy, IxIy, fsize)
    end = time.perf_counter()  # 시간 측정 끝
    print('M_harris time : ', end-start)

    R = np.zeros((h, w))
    for row in range(h):
        for col in range(w):
            ###########################################################################
            # ToDo
            # det_M 계산
            # trace_M 계산
            # R 계산 Harris & Stephens (1988), Nobel (1998) 어떤걸로 구현해도 상관없음
            ###########################################################################
            det_M = M_harris[row, col, 0, 0]*M_harris[row, col, 1, 1]-M_harris[row, col, 0, 1]*M_harris[row, col, 1, 0]
            trace_M = M_harris[row, col, 0, 0] + M_harris[row, col, 1, 1]
            R[row, col] = det_M - (k * (trace_M ** 2))
            # R[row, col] = det_M/(trace_M+(1e+8))

    # thresholding
    R[R < threshold_rate * np.max(R)] = 0

    R = find_local_maxima(R, 21)
    R = cv2.dilate(R, None)

    harris_img[R != 0]=[0, 0, 255]

    return harris_img
```

* harris_detector_integral

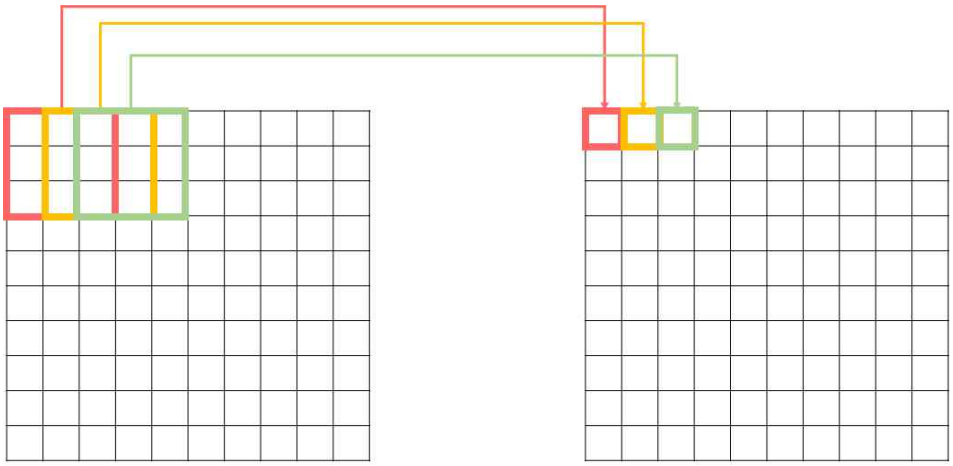
```python
    ############################
    # ToDo
    # M_integral 완성시키기
    ############################
    h, w = IxIx_integral.shape
    M_integral = np.zeros((h, w, 2, 2))
    for row in range(h-(fsize-1)):
        for col in range(w-(fsize-1)):
            if row == 0 and col == 0:
                M_integral[row, col, 0, 0] = IxIx_integral[row + (fsize-1), col + (fsize-1)]
                M_integral[row, col, 0, 1] = IxIy_integral[row + (fsize-1), col + (fsize-1)]
                M_integral[row, col, 1, 0] = M_integral[row, col, 0, 1]
                M_integral[row, col, 1, 1] = IyIy_integral[row + (fsize-1), col + (fsize-1)]
            elif row == 0:
                M_integral[row, col, 0, 0] = IxIx_integral[row + (fsize-1), col + (fsize-1)] - IxIx_integral[row + (fsize-1), col - 1]
                M_integral[row, col, 0, 1] = IxIy_integral[row + (fsize-1), col + (fsize-1)] - IxIy_integral[row + (fsize-1), col -1]
                M_integral[row, col, 1, 0] = M_integral[row, col, 0, 1]
                M_integral[row, col, 1, 1] = IyIy_integral[row + (fsize-1), col + (fsize-1)] - IyIy_integral[row + (fsize-1), col - 1]
            elif col == 0:
                M_integral[row, col, 0, 0] = IxIx_integral[row + (fsize-1), col + (fsize-1)] - IxIx_integral[row -1, col +(fsize-1)]
                M_integral[row, col, 0, 1] = IxIy_integral[row + (fsize-1), col + (fsize-1)] - IxIy_integral[row -1, col +(fsize-1)]
                M_integral[row, col, 1, 0] = M_integral[row, col, 0, 1]
                M_integral[row, col, 1, 1] = IyIy_integral[row + (fsize-1), col + (fsize-1)] - IyIy_integral[row -1, col +(fsize-1)]
            else:
                M_integral[row, col, 0, 0] = IxIx_integral[row - 1, col-1] + IxIx_integral[row +(fsize-1), col +(fsize-1)] - IxIx_integral[row-1, col +(fsize-1)] - IxIx_integral[row+(fsize-1), col-1]
                M_integral[row, col, 0, 1] = IxIy_integral[row - 1, col-1] + IxIy_integral[row +(fsize-1), col +(fsize-1)] - IxIy_integral[row-1, col +(fsize-1)] - IxIy_integral[row+(fsize-1), col-1]
                M_integral[row, col, 1, 0] = M_integral[row, col, 0, 1]
                M_integral[row, col, 1, 0] = IyIy_integral[row - 1, col-1] + IyIy_integral[row +(fsize-1), col +(fsize-1)] - IyIy_integral[row-1, col +(fsize-1)] - IyIy_integral[row+(fsize-1), col-1]

    end = time.perf_counter()  # 시간 측정 끝
    print('M_harris integral time : ', end-start)
```

```
end = time.perf_counter()  # 시간 측정 끝
print('M_harris integral time : ', end-start)

R = np.zeros((h, w))
for row in range(h):
    for col in range(w):
        ########################################################################
        # ToDo
        # det_M 계산
        # trace_M 계산
        # R 계산 Harris & Stephens (1988), Nobel (1998) 어떤걸로 구현해도 상관없음
        ########################################################################
        det_M = M_integral[row, col, 0, 0] * M_integral[row, col, 1, 1] - M_integral[row, col, 0, 1] * M_integral[row, col, 1, 0]
        trace_M = M_integral[row, col, 0, 0] + M_integral[row, col, 1, 1];
        R[row, col] = det_M - (k * (trace_M ** 2))

# thresholding
R[R < threshold_rate * np.max(R)] = 0

R = find_local_maxima(R, 21)
R = cv2.dilate(R, None)

harris_img[R != 0]=[0, 0, 255]

return harris_img
```

2. 코드 설명

* calc_M_harris

만약 필터의 크기가 3이라고 가정할 때 M은 아래와 같이 구한다. IxIx, IxIy, IyIy 각각에서 필터 안의 값들을 모두 더해 M을 채운다. calc_M_harris는 필터 안의 값들을 모두 더하는 과정을 4중 for문을 통해 구현했다.



* harris_detector

$R = \det(M) - ktrace^2(M)$ 공식을 구현하기 위해 det(M)과 trace(M)을 구한다.

* harris_detector_integral

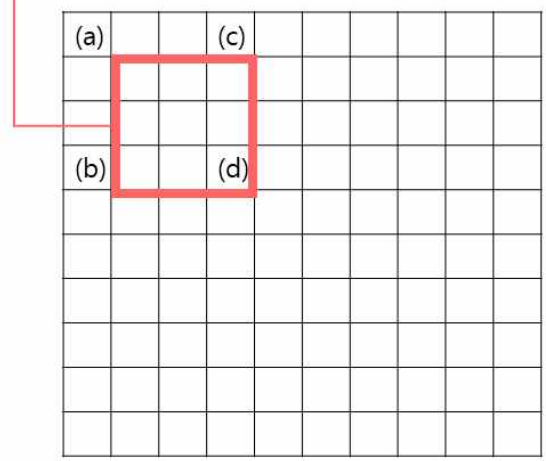
integral image를 이용하면 필터 안의 값들을 모두 더하는 연산을 반복하지 않아도 된다. integral image는 아래와 같은 방법으로 만들었다.

d 위치의 integral 값 = c까지의 integral 값+
(b까지의 integral 값 – a까지의 integral 값) +
d의 original 값

31(a)	2(b)	4	33	5
36(c)	12(d)	26	9	10
29	25	13	17	21
22	20	18	24	23
12	16	14	19	17

M_integral은 필터 안의 값들을 하나하나 다 더하지 않고 IxIx, IxIy, IyIy 각각의 integral 이미지로 적분 특성을 이용해 아래와 같은 방법으로 구한다. 적분 특성을 이용하면 3번의 연산으로 필터 안 값들의 합을 구할 수 있다.

빨간 영역의 합 = (a)+(d)-(c)-(b)



IxIx, IxIy, IyIy의 integral image

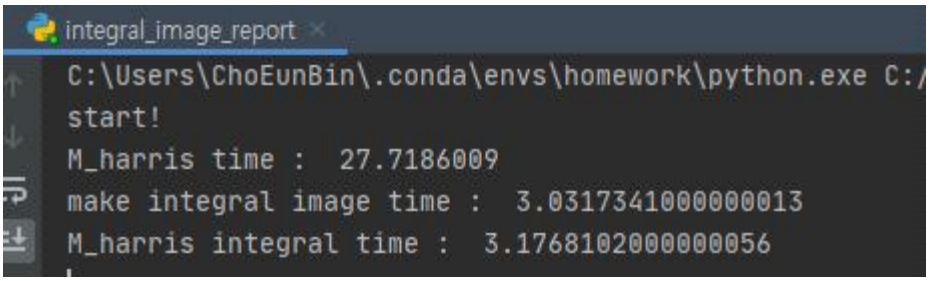
3. 이미지

* Orignal

* Harris corner

* Harris corner integral image

* 결과



4. 느낀점

이론 수업을 듣는 것만으로 실습을 진행하기에는 조금 어려운 감이 있었습니다.

5. 과제 난이도

조금 어려웠습니다.