Image Processing
실습 10주차

김 대 현

Department of Computer Science and Engineering

Chungnam National University, Korea

실습 소개

- **과목 홈페이지**
  - 충남대학교 사이버 캠퍼스 (https://dcs-lcms.cnu.ac.kr/login?redirectUrl=https://dcs-lcms.cnu.ac.kr/)
- **TA 연락처**
  - 김대현
  - 공대 5호관 531호 컴퓨터비전 연구실
  - Email: seven776484@gmail.com
    - [IP]을 이메일 제목에 붙여주세요
    - 과제 질문은 메일 또는 사전에 미리 연락하고 연구실 방문 가능
- **Tutor 연락처**
  - 정주헌
  - Email: 201802015@o.cnu.ac.kr

목차

- **공지사항**
  - 9주차 과제(Otsu’s method) 구현 강조사항
  - 영상처리 02· 03분반 4주차 Filtering 과제 채점 기준
  - 6주차 과제(DoG) 리뷰
- **과제**
  - DCT(Discrete cosine transform) mask 완성

공지사항

- **9주차 과제(Otsu's method) 구현 강조사항**
  - Inter variance로 구현 시 반드시 *Moving average로 구현*
  - Moving average로 구현 안 할 시 감점

```python
def get_threshold_by_within_variance(intensity, p):

    """
    :param intensity: pixel 값 0 ~ 255 범위를 갖는 배열
    :param p: 상대도수 값
    :return: k: 최적의 threshold 값
    """

    ############################################################
    # TODO
    # TODO otsu_method 완성
    # TODO  1. within-class variance를 이용한 방법
    # TODO  교수님 이론 PPT 22 page 참고
    ############################################################

    ???

    return k
```

```python
def get_threshold_by_inter_variance(p):

    """
    :param p: 상대도수 값
    :return: k: 최적의 threshold 값
    """

    ############################################################
    # TODO
    # TODO otsu_method 완성
    # TODO  2. inter-class variance를 이용한 방법
    # TODO  Moving average를 이용하여 구현
    # TODO  교수님 이론 PPT 26 page 참고
    ############################################################

    p += 1e-7  # q1과 q2가 0일때 나눗셈을 진행할 경우 오류를 막기 위함

    ???

    return k
```

공지사항

- **영상처리 02· 03분반 4주차 과제 채점 기준**
  - 4주차(filtering) 채점 기준
    - 코드에서는 다음과 같이 3개의 과제 구현 코드 평가
      - filtering_analysis
      - Gaussian_analysis1
      - Gaussian_analysis2

공지사항

- **영상처리 02· 03분반 4주차 과제 채점 기준**
  - 4주차(filtering) 채점 기준
    - 기본적으로 과제 코드는 02 · 03분반 동일
    - Filtering analysis
      - 다음과 같이 2개의 함수 평가
      - 함수 한 개당 잘못 구현 시 1점 감점(부분 점수 없음)

```python
def generate_average_filter(ksize):
    """
    인자 정보
    ksize: 커널 크기

    return kernel
    """
    ############################################################
    # TODO                                                     #
    # average filter 구현                                      #
    ############################################################

    kernel = ???
    return kernel
```

```python
    ############################################################
    # TODO                                                     #
    # Filtering 구현                                           #
    # dst : filtering 결과 image                               #
    # 유의미한 시간 측정을 위해 4중 for 문으로 구현 할 것      #
    # 교수님 이론 PPT 4page 수식 참고                          #
    ############################################################

    ???

    # float32 -> uint8(unsigned int)로 변경
    dst = np.round(dst).astype(np.uint8)
    return dst
```

공지사항

- **영상처리 02· 03분반 4주차 과제 채점 기준**
  - 4주차(filtering) 채점 기준
    - Gaussian analysis1
      - 다음과 같이 3개의 함수 평가
      - 함수 한 개당 잘못 구현 시 기본적으로 1점 감점(my_get_Gaussian_filter를 제외한 나머지 함수 부분 점수 없음)
      - my_get_Gaussian_filter의 경우 주석과 같이 y, x를 2차원으로 구현 시 0.5점 감점

```python
def my_filtering(src, mask, pad_type='zero'):
    (h, w) = src.shape
    (f_h, f_w) = mask.shape
    pad_img = my_padding(src, mask, pad_type)
    dst = np.zeros((h, w))

    ########################################
    # TODO 3. Filtering 2중 for문 구현
    ########################################

    for row in range(h):
        for col in range(w):
            dst = ???
    dst = np.round(dst).astype(np.uint8)
    return dst
```

```python
################################################
# TODO 1. sigma 변화에 따른 1D Gaussian filter 시각화
# TODO gaussian_filter 식 채우기 및 그래프 그리기
# TODO 교수님 이론 PPT 40 page 참고
################################################
x = np.linspace(-5, 5, 1000)
mean = x.mean()
sigma1 = 0.5

gaussian_filter1 = ???

sigma2 = 1
gaussian_filter2 = ???

sigma3 = 2
gaussian_filter3 = ???
```

```python
def my_get_Gaussian_filter(fshape, sigma=1):

    (f_h, f_w) = fshape

    ################################################
    # TODO 2. 1D Gaussian으로 2번 Filtering
    # TODO gaussian_filter 식 채우기
    # TODO 참고) np.mgrid 사용하면 쉽게 구현 가능
    # TODO hint
    #     y, x = np.mgrid[-1:2, -1:2]
    #     y => [[-1,-1,-1],
    #           [ 0, 0, 0],
    #           [ 1, 1, 1]]
    #     x => [[-1, 0, 1],
    #           [-1, 0, 1],
    #           [-1, 0, 1]]
    ################################################
```

공지사항

- **영상처리 02· 03분반 4주차 과제 채점 기준**
  - 4주차(filtering) 채점 기준
    - Gaussian analysis2
      - 다음과 같이 2개의 함수 중 2D Gaussian 함수 평가(왼쪽 함수는 중복돼서 제외)
      - 함수 잘못 구현 시 2점 감점(부분 점수 없음)

```python
def my_filtering(src, mask, pad_type='zero'):
    (h, w) = src.shape
    (f_h, f_w) = mask.shape
    pad_img = my_padding(src, mask, pad_type)
    dst = np.zeros((h, w))

    ########################################
    # TODO 3. Filtering 2중 for문 구현
    ########################################

    for row in range(h):
        for col in range(w):
            dst = ???
    dst = np.round(dst).astype(np.uint8)
    return dst
```

```python
def my_get_Gaussian_filter(fshape, sigma=1):

    (f_h, f_w) = fshape
    ########################################################
    # TODO 2 2D Gaussian filter 구현
    # TODO 2 np.mrid를 사용하면 y, x 모두 구할 수 있음
    # TODO hint
    #     y, x = np.mgrid[-1:2, -1:2]
    #     y => [[-1,-1,-1],
    #           [ 0, 0, 0],
    #           [ 1, 1, 1]]
    #     x => [[-1, 0, 1],
    #           [-1, 0, 1],
    #           [-1, 0, 1]]
    ########################################################
```

공지사항

- **영상처리 02· 03분반 4주차 과제 채점 기준**
  - 4주차(filtering) 채점 기준
    - 보고서 기준
      - Filtering analysis 과제의 경우 *시간 측정 비교 내용이 간단히 언급만 되어있으면 감점 없음*
      - 해당 내용이 없으면 1점 감점

- **커널 크기에 따른 속도 비교**
  - Average filter를 사용하여 다양한 커널 크기로 이미지를 filtering한 시간 측정
  - 본 과제에서는 kernel 크기를 3×3, 7×7, 15×15로 설정
  - 보고서에 각 크기에 따른 시간 측정 기록 및 시간 차이가 나는 이유에 대한 정량적 분석 서술
  - 정량적 분석은 **filtering 연산량을 기입**할 것(이미지의 높이와 너비, 커널 크기를 활용)
  - **시간은 엄연히 하드웨어에 영향을 받기 때문에 학생들 마다 다를 수 있다**

02분반

- **Filtering 시간 체크**
  - 커널의 크기에 따라 연산 수행 시간 비교
  - Ksize = 3, 7, 15로 비교
  - 시간 측정 및 이미지 분석

03분반

공지사항

- **영상처리 02 · 03분반 4주차 과제 채점 기준**
  - 4주차(filtering) 채점 기준
    - 보고서 기준
      - Gaussian filtering 과제의 경우 각 case에 대해서 커널 크기와 sigma에 따른 간단한 분석이 있을 경우 감점 없음 (각 case는 02 · 03분반 모두 동일)
      - 해당 내용이 없으면 1점 감점

- **Gaussian filtering 분석**
  - Gaussian filter의 커널 크기와 시그마에 따라 filtering된 이미지가 어떻게 변화하는지에 대한 분석
  - 커널 크기와 시그마에 따른 총 6가지 경우(Gaussian 2D에 한함)에 대한 filtering된 결과를 커널 시각화 이미지와 Gaussian 1D 그래프를 바탕으로 해석한 내용을 보고서에 서술
  - 보고서 첨부 자료(중요)
    - 다양한 시그마에 크기에 따른 Gaussian 1D 그래프
    - 커널 크기와 시그마에 따른 6가지 경우에 대한 filtering된 결과 이미지 및 커널 시각화 이미지

02분반

- **Gaussian filtering 분석**
  - Gaussian 1D, 2D 구현
  - Ksize, sigma에 따라 이미지가 어떻게 변하는지와 이유에 대해서 분석
  - Matplotlib 라이브러리를 활용하여 필터를 시각화하면 좋음.
  - Case
    1) ksize = 5 x 5 sigma = 1
    2) ksize = 5 x 5 sigma = 3
    3) ksize = 5 x 5 sigma = 0.1
    4) ksize = 7 x 7 sigma = 3
    5) ksize = 11 x 11 sigma = 3
    6) ksize = 15 x 15 sigma = 3

03분반

DoG 과제 리뷰

- **Filtering을 이용한 DoG 필터 마스크 구현**
  - 절차
    - 1. 각 축 방향의 크기 3을 갖는 1차원 미분 vector를 생성

```
In[2]: derivate_x
Out[2]: array([[-1,  0,  1]])
```

```
In[3]: derivate_y
Out[3]:
array([[-1],
       [ 0],
       [ 1]])
```

    - 2. 미분 vector의 크기에 맞게 Gaussian filter를 확장하여 생성
      -> 미분 vector 크기가 3이므로 (kernel_size + 2) x (kernel_size + 2) 만큼 확장
      -> 또는 각 축 방향의 filtering을 고려해서 각각 (kernel_size + 2) x (kernel_size) 와 (kernel_size) x (kernel_size + 2) 크기를 갖는 Gaussian filter 따로 생성

DoG 과제 리뷰

- **Filtering을 이용한 DoG 필터 마스크 구현**
  - 절차
    - 3. 각 축 방향에 대해서 미분 Filtering
      - X축 미분(5 x 5 gaussian filter 예시)
        - 7 × 7 또는 5 × 7로 확장*(padding 적용 안됨)*

-1	0	1	3	2	1	
	6	7	6	4	3	
	7	9	8	7	5	
	4	6	7	8	6	
	1	2	3	4	3	

$$f'(x) = \frac{f(x+1) - f(x-1)}{2}$$

교수님 이론 PPT 12 Page

DoG 과제 리뷰

- **Filtering을 이용한 DoG 필터 마스크 구현**
  - 절차
    - 3. 각 축 방향에 대해서 미분 Filtering
      - X축 미분(5 x 5 gaussian filter 예시)
        - 7 × 7 또는 5 × 7로 확장*(padding 적용 안됨)*

	-1	0	1	2	1	
	6	7	6	4	3	
	7	9	8	7	5	
	4	6	7	8	6	
	1	2	3	4	3	

$$f'(x) = \frac{f(x+1) - f(x-1)}{2}$$

교수님 이론 PPT 12 Page

DoG 과제 리뷰

- **Filtering을 이용한 DoG 필터 마스크 구현**
  - 절차
    - 3. 각 축 방향에 대해서 미분 Filtering
      - X축 미분(5 x 5 gaussian filter 예시)
        - 7 × 7 또는 5 × 7로 확장*(padding 적용 안됨)*

	1	2	3	2	1	
	6	7	6	4	3	
	7	9	8	7	5	
	4	6	7	8	6	
	1	2	3	-1	0	1

$$f'(x) = \frac{f(x+1) - f(x-1)}{2}$$

교수님 이론 PPT 12 Page

DoG 과제 리뷰

- **Filtering을 이용한 DoG 필터 마스크 구현**
  - 절차
    - 3. 각 축 방향에 대해서 미분 Filtering
      - y축 미분(5 x 5 gaussian filter 예시) filtering
        - 7×7 또는 7×5로 확장*(padding 적용 안됨)*

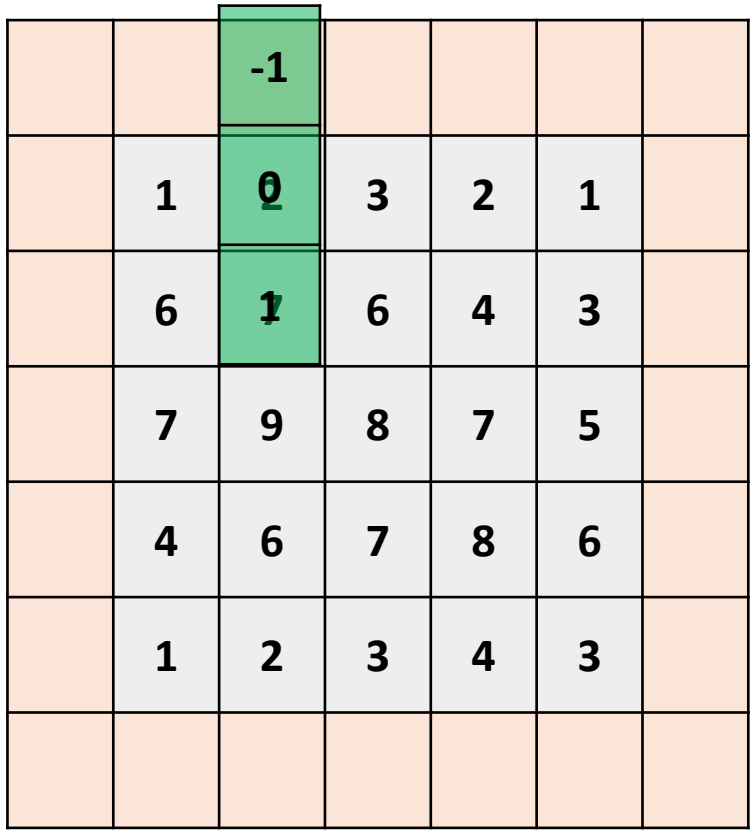
	-1					
	0	2	3	2	1	
	1	7	6	4	3	
	7	9	8	7	5	
	4	6	7	8	6	
	1	2	3	4	3	

$$f'(x) = \frac{f(x+1) - f(x-1)}{2}$$

교수님 이론 PPT 12 Page

DoG 과제 리뷰

- **Filtering을 이용한 DoG 필터 마스크 구현**
  - 절차
    - 3. 각 축 방향에 대해서 미분 Filtering
      - y축 미분(5 x 5 gaussian filter 예시) filtering
        - 7×7 또는 7×5로 확장*(padding 적용 안됨)*



$$f'(x) = \frac{f(x+1) - f(x-1)}{2}$$

교수님 이론 PPT 12 Page

DoG 과제 리뷰

- **Filtering을 이용한 DoG 필터 마스크 구현**
  - 절차
    - 3. 각 축 방향에 대해서 미분 Filtering
      - y축 미분(5 x 5 gaussian filter 예시) filtering
        - 7×7 또는 7×5로 확장*(padding 적용 안됨)*

					-1	
	1	2	3	2	0	
	6	7	6	4	1	
	7	9	8	7	5	
	4	6	7	8	6	
	1	2	3	4	3	

$$f'(x) = \frac{f(x+1) - f(x-1)}{2}$$

교수님 이론 PPT 12 Page

DoG 과제 리뷰

- **Filtering을 이용한 DoG 필터 마스크 구현**
  - 절차
    - 3. 각 축 방향에 대해서 미분 Filtering
      - y축 미분(5 x 5 gaussian filter 예시) filtering
        - 7×7 또는 7×5로 확장*(padding 적용 안됨)*

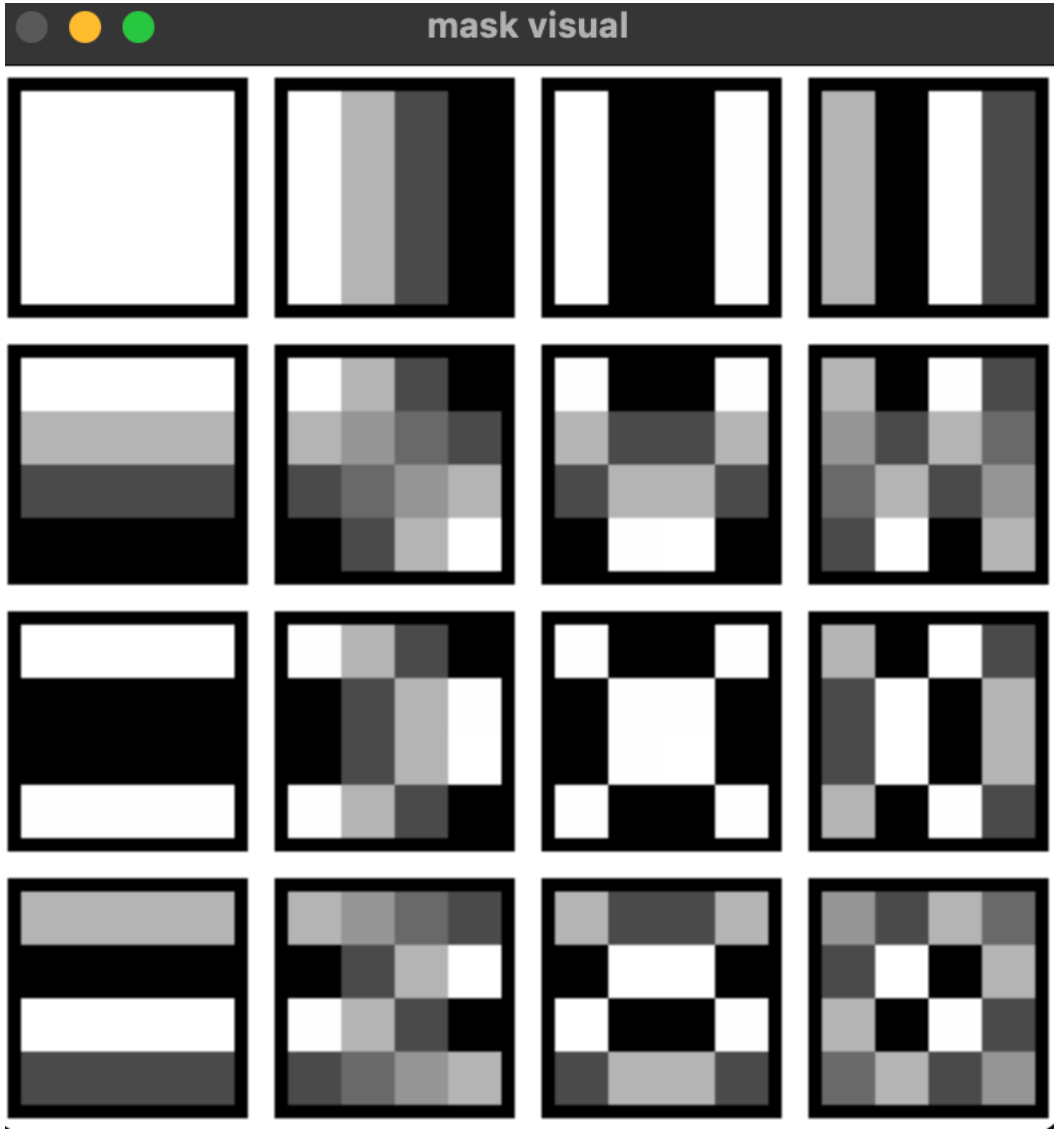
	1	2	3	2	1	
	6	7	6	4	3	
	7	9	8	7	5	
	4	6	7	8	-1	
	1	2	3	4	0	
					1	

$$f'(x) = \frac{f(x+1) - f(x-1)}{2}$$

교수님 이론 PPT 12 Page

과제

- **DCT(Discrete cosine transform) mask 완성**

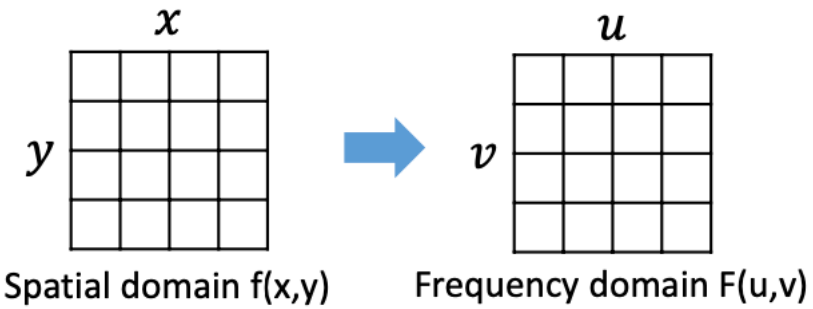


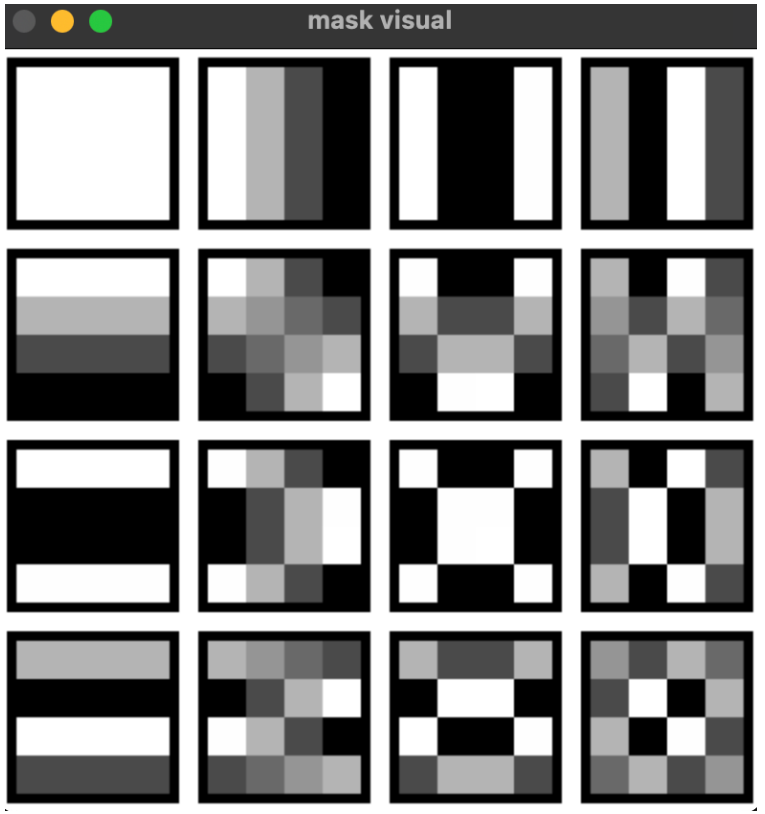
과제

- **DCT(Discrete cosine transform) mask 완성**
  - 4×4의 크기를 갖는 Filter mask 생성
  - 4×4의 크기를 갖는 Mask가 총 16개 (16×16)

$$F(u,v) = C(u)C(v)\sum_{y=0}^{n-1}\sum_{x=0}^{n-1} f(x,y)\cos\left(\frac{(2x+1)u\pi}{2n}\right)\cos\left(\frac{(2y+1)v\pi}{2n}\right)$$

$$C(w) = \begin{cases} \sqrt{1/n} & if\ w = 0 \\ \sqrt{2/n} & otherwise \end{cases}$$



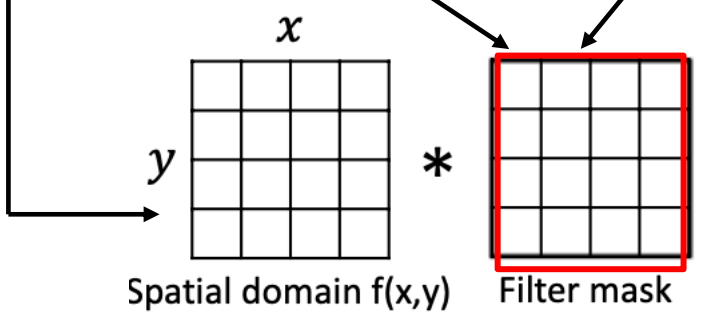


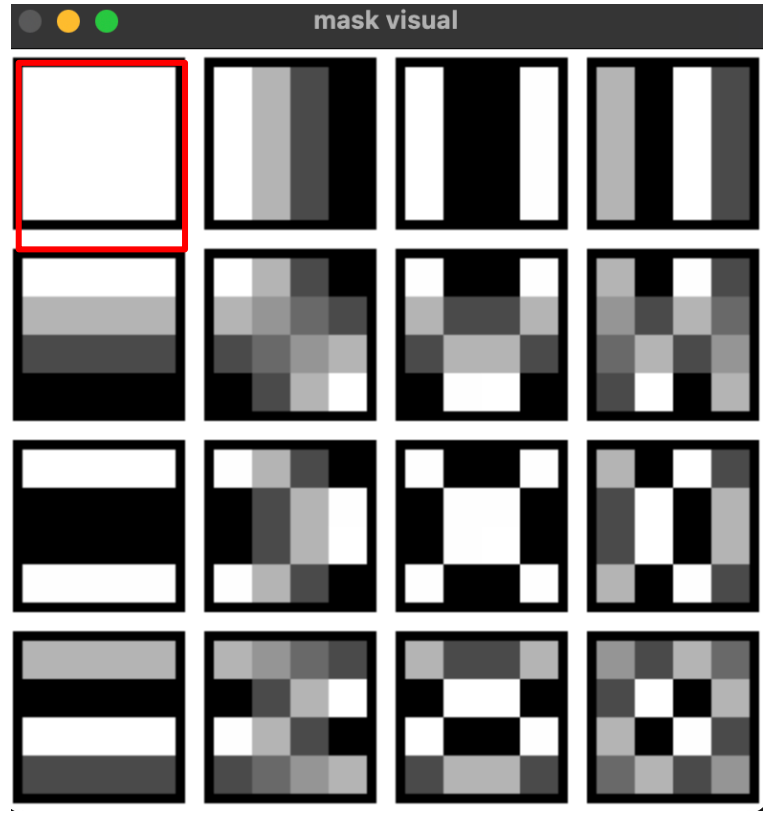
과제

- **DCT(Discrete cosine transform) mask 완성**

$$F(0,0) = C(0)C(0)\sum_{y=0}^{3}\sum_{x=0}^{3} f(x,y)\cos(0)\cos(0)$$

$$F(0,0) = C(0)C(0)\sum_{y=0}^{3}\sum_{x=0}^{3} f(x,y)\cos(0)\cos(0)$$



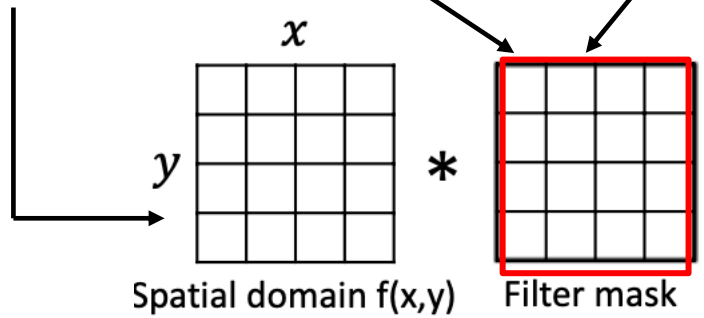


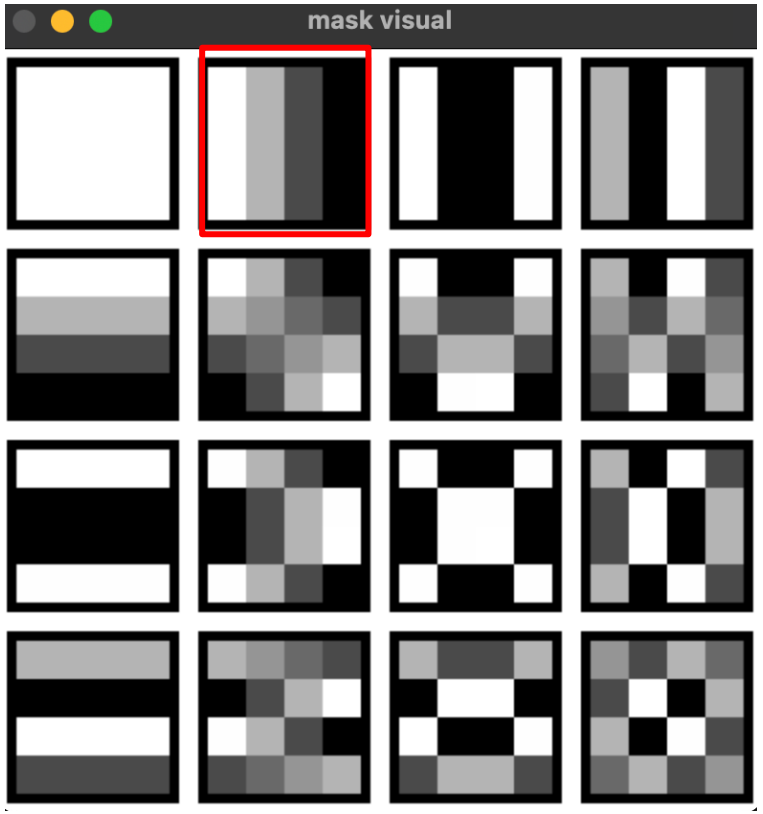
과제

- **DCT(Discrete cosine transform) mask 완성**

$$F(1,0)$$

$$= C(1)C(0)\sum_{y=0}^{3}\sum_{x=0}^{3} f(x,y)\cos\left(\frac{(2x+1)\pi}{2n}\right)\cos(0)$$





과제

- **DCT(Discrete cosine transform) mask 완성**
  - 구현 상세
    - 4×4로 이루어진 $Sub\ mask$를 총 16개(16×16)형태로 생성

```
for v_ in range(v):
    for u_ in range(u):

        ################################################################################
        # TODO
        # TODO mask 만들기
        # TODO sub mask shape : 4 x 4
        # TODO full mask shape = 16 x 16
        # TODO DCT에서 사용된 mask는 4 x 4 mask가 16개 있음 (u, v) 별로 1개씩 있음 u=4, v=4
        # TODO submask 마다 0 ~ 255의 범위를 갖도록 변환 (my_transform 함수 사용)
        # TODO full mask는 각 sub mask로 구성되어있음
        ################################################################################
        submask = ???

        # normalization

return full_mask
```

과제

- **DCT(Discrete cosine transform) mask 완성**
  - 구현 상세
    - 4×4로 이루어진 $Sub\ mask$를 [0,1] 범위로 Normalization 후 0 ~ 255의 값을 갖도록 변환

```
def my_transform(src):
    """

    :param src: sub mask
    :return: dst
    """

    ########################################################################
    # TODO
    # TODO my_normalize
    # TODO mask를 normalization(0 ~ 1)후 (0 ~ 255)의 값을 갖도록 변환
    ########################################################################
    ???

    return dst
```

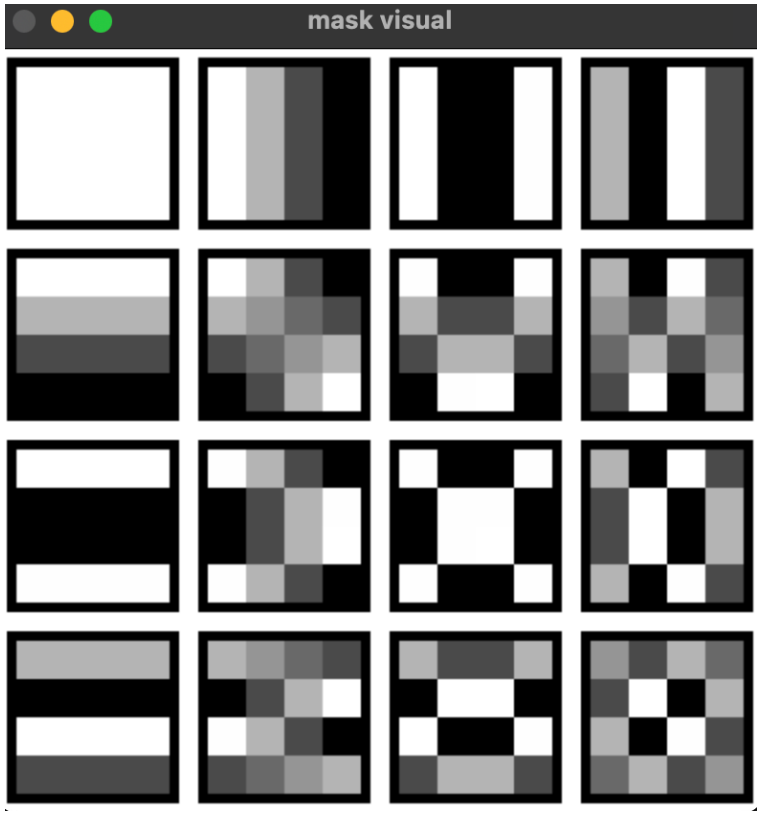
과제

- **DCT(Discrete cosine transform) mask 완성**
  - 구현 상세
    - 정답 mask와 비교 시 True 값이 출력되어야 함

```
transform mask :
[[255 255 255 255 255 180  74   0 255   0   0 254 180   0 255  74]
 [255 255 255 255 255 180  74   0 255   0   0 254 180   0 255  74]
 [255 255 255 255 255 180  74   0 255   0   0 254 180   0 255  74]
 [255 255 255 255 255 180  74   0 255   0   0 254 180   0 255  74]
 [255 255 255 255 255 180  74   0 254   0   0 254 180   0 254  74]
 [180 180 180 180 180 149 105  74 180  74  74 180 149  74 180 105]
 [ 74  74  74  74  74 105 149 180  74 180 180  74 105 180  74 149]
 [  0   0   0   0   0  74 180 255   0 254 255   0  74 255   0 180]
 [255 255 255 255 254 180  74   0 254   0   0 254 180   0 255  74]
 [  0   0   0   0   0  74 180 254   0 254 254   0  74 255   0 180]
 [  0   0   0   0   0  74 180 255   0 254 255   0  74 255   0 180]
 [254 254 254 254 254 180  74   0 254   0   0 254 180   0 254  74]
 [180 180 180 180 180 149 105  74 180  74  74 180 149  74 180 105]
 [  0   0   0   0   0  74 180 255   0 255 255   0  74 255   0 180]
 [255 255 255 255 254 180  74   0 255   0   0 254 180   0 254  74]
 [ 74  74  74  74  74 105 149 180  74 180 180  74 105 180  74 149]]
```

과제

- **DCT(Discrete cosine transform) mask 완성**
  - 보고서 사진 첨부 목록
    - mask visual(오른쪽 이미지) 첨부
  - 과제 진행 2023.5월 05일 ~ 2023년 5월 11일 23:59



Q & A