Image Processing
실습 12주차

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

공지사항

- 공지사항
- Canny Edge 과제 리뷰
- 실습
- 과제 (Bilateral filtering 구현)

목차

- **공지사항**
  - 영상처리 02 · 03분반 과제 채점 공개
    - 이번주 중으로 Canny Edge Detection 과제 채점 진행
    - 채점은 기본적으로 각 주차 별 채점 기준표에 의거하여 각 분반 조교가 채점
    - 과제 채점에 대한 문의 사항은 각 분반 조교에게 문의
  - 영상처리 02 · 03분반 과제 채점 기준 공개
    - Canny Edge Detection 채점 기준 공개
  - 과제 Copy 관련 공지
    - 과제 Copy와 관련하여 과제를 함께 진행하였을 경우 보고서에 같이 과제를 진행한 학부생 학번 기입
  - 과제 점수 문의는 메일 또는 수업시간 이후 질문시간을 이용.

Canny Edge 과제 리뷰

- **Canny Edge 과제 리뷰**
  - 다음과 같은 수식을 통해 Magnitude랑 Angle을 구함.

- **2D gradient of an image:**

$$\nabla I = (I_x, I_y) = \left(\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y} \right)$$

- **The gradient magnitude (edge strength):**

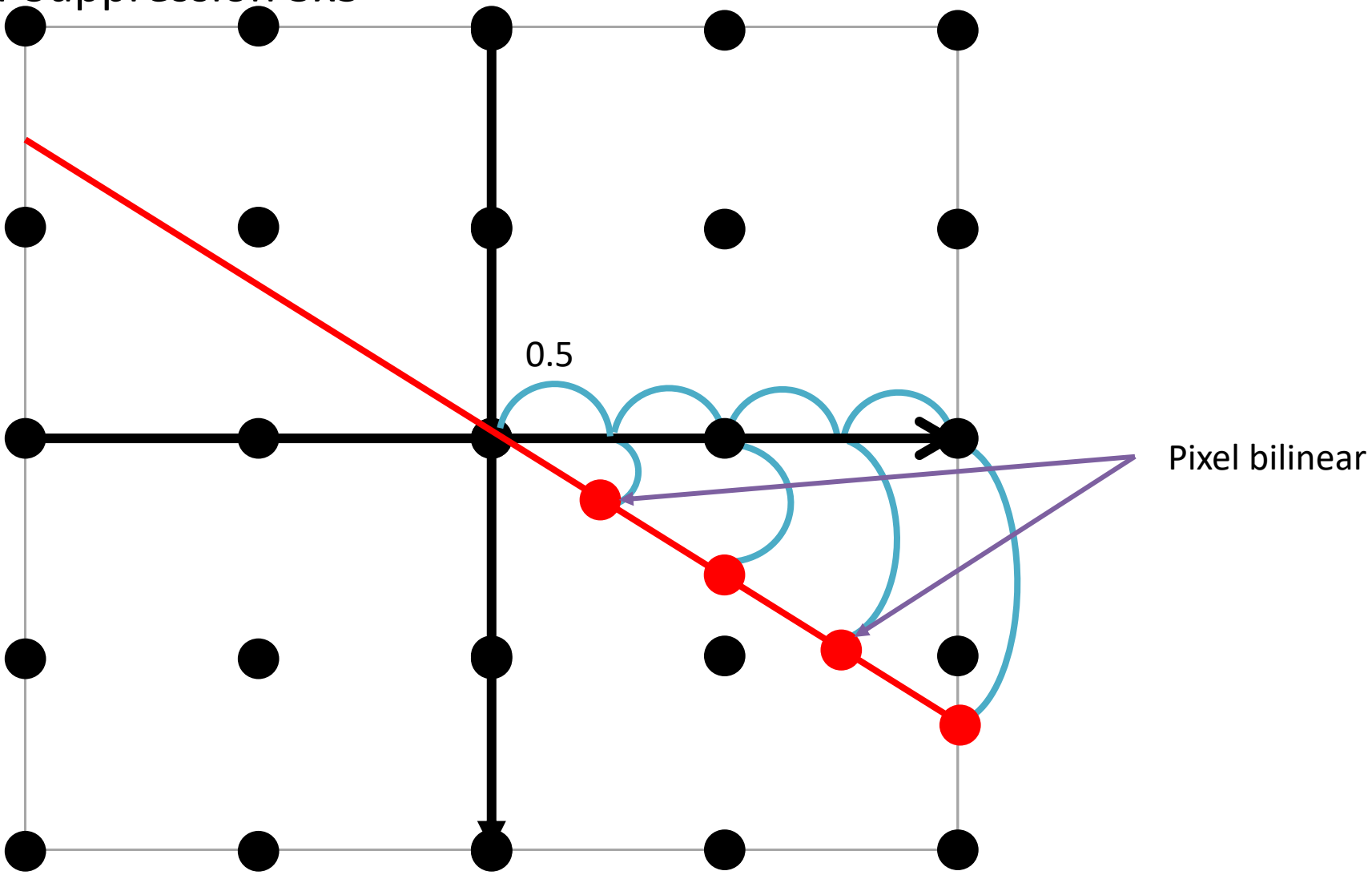
$$\|\nabla I\| = \sqrt{I_x^2 + I_y^2}$$

- **The gradient direction:**

$$\theta = \tan^{-1}\left(\frac{I_y}{I_x} \right)$$

Canny Edge 과제 리뷰

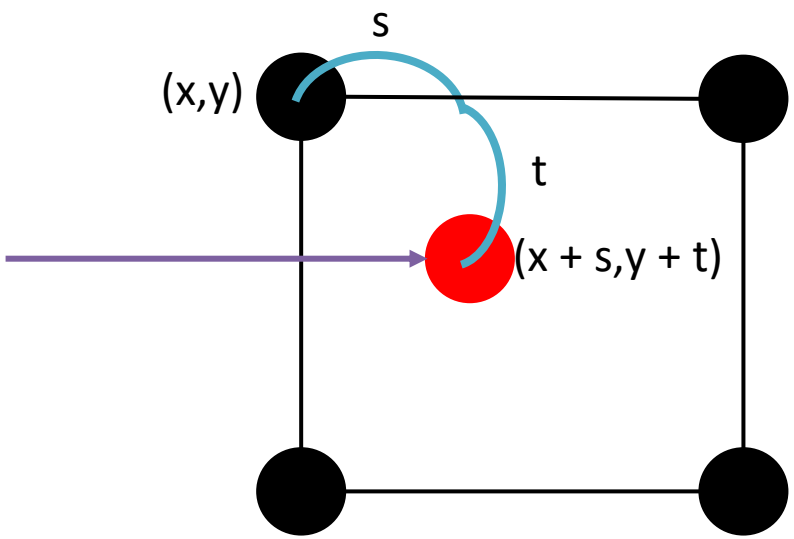
- **Canny Edge 과제 리뷰**
  - Non-Maximum-Suppression 5x5



Canny Edge 과제 리뷰

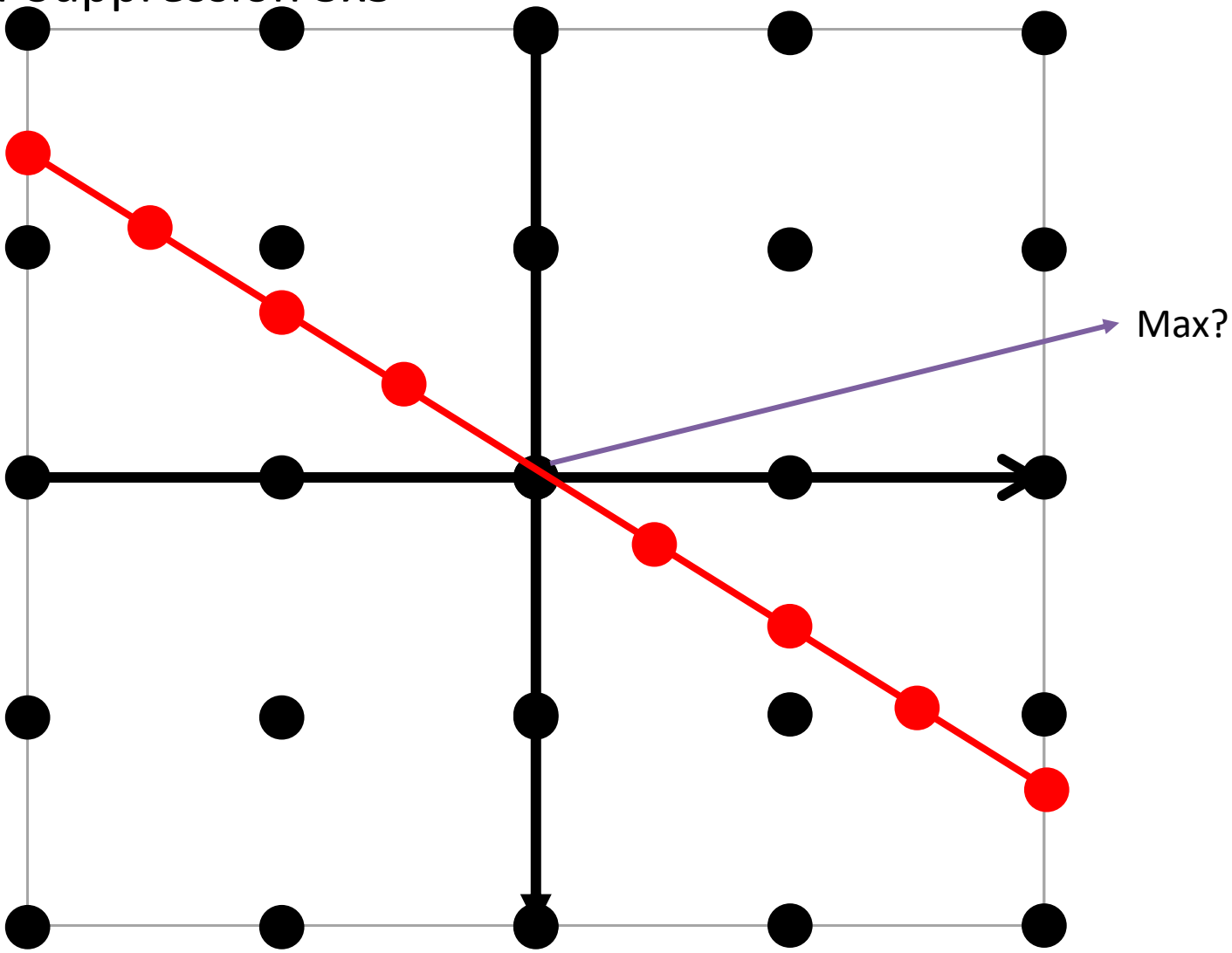
- **Canny Edge 과제 리뷰**
  - Pixel bilinear

빨간 점을 통해서 주변 픽셀 4개를 가져올 수 있음



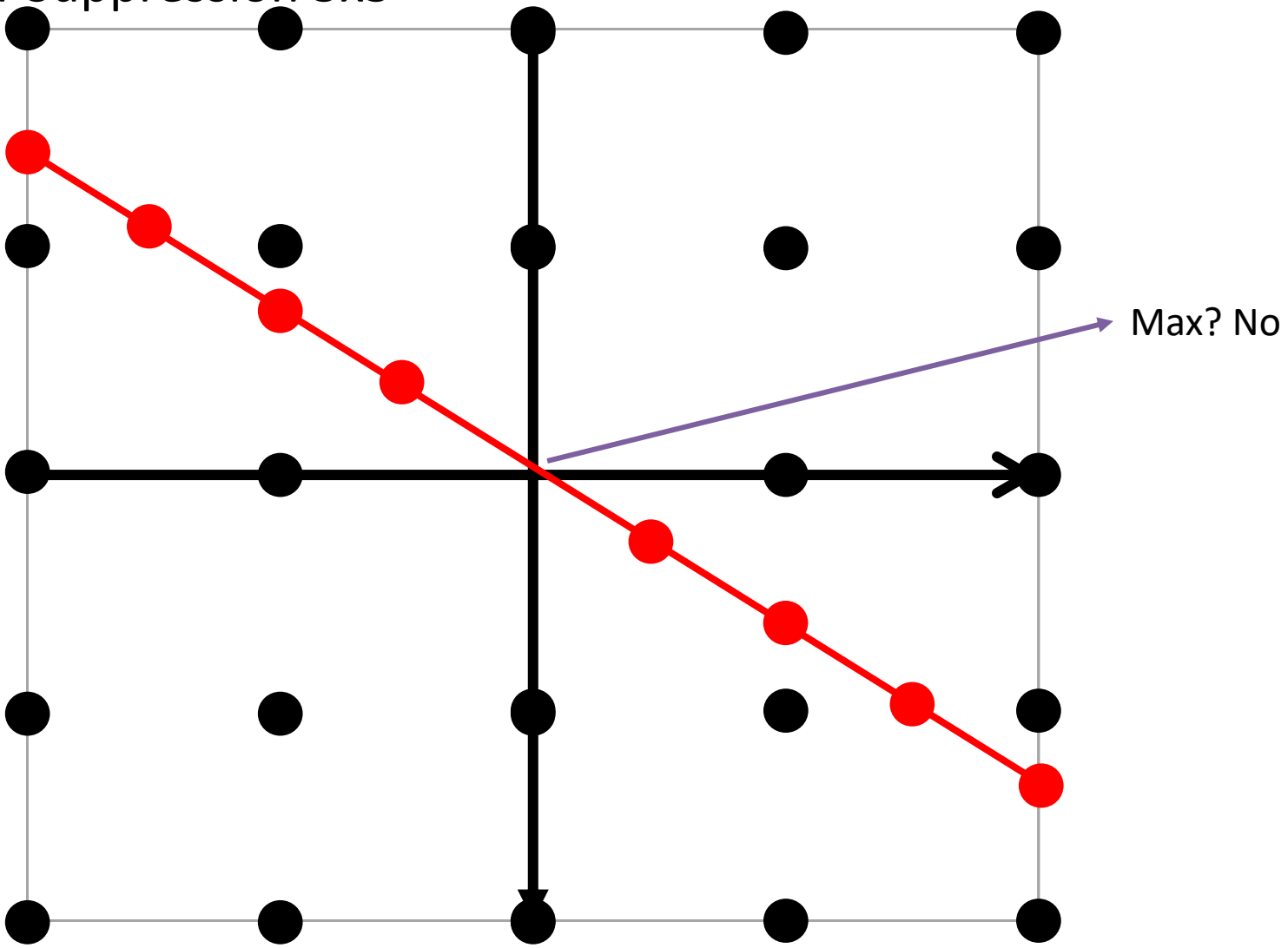
Canny Edge 과제 리뷰

- **Canny Edge 과제 리뷰**
  - Non-Maximum-Suppression 5x5



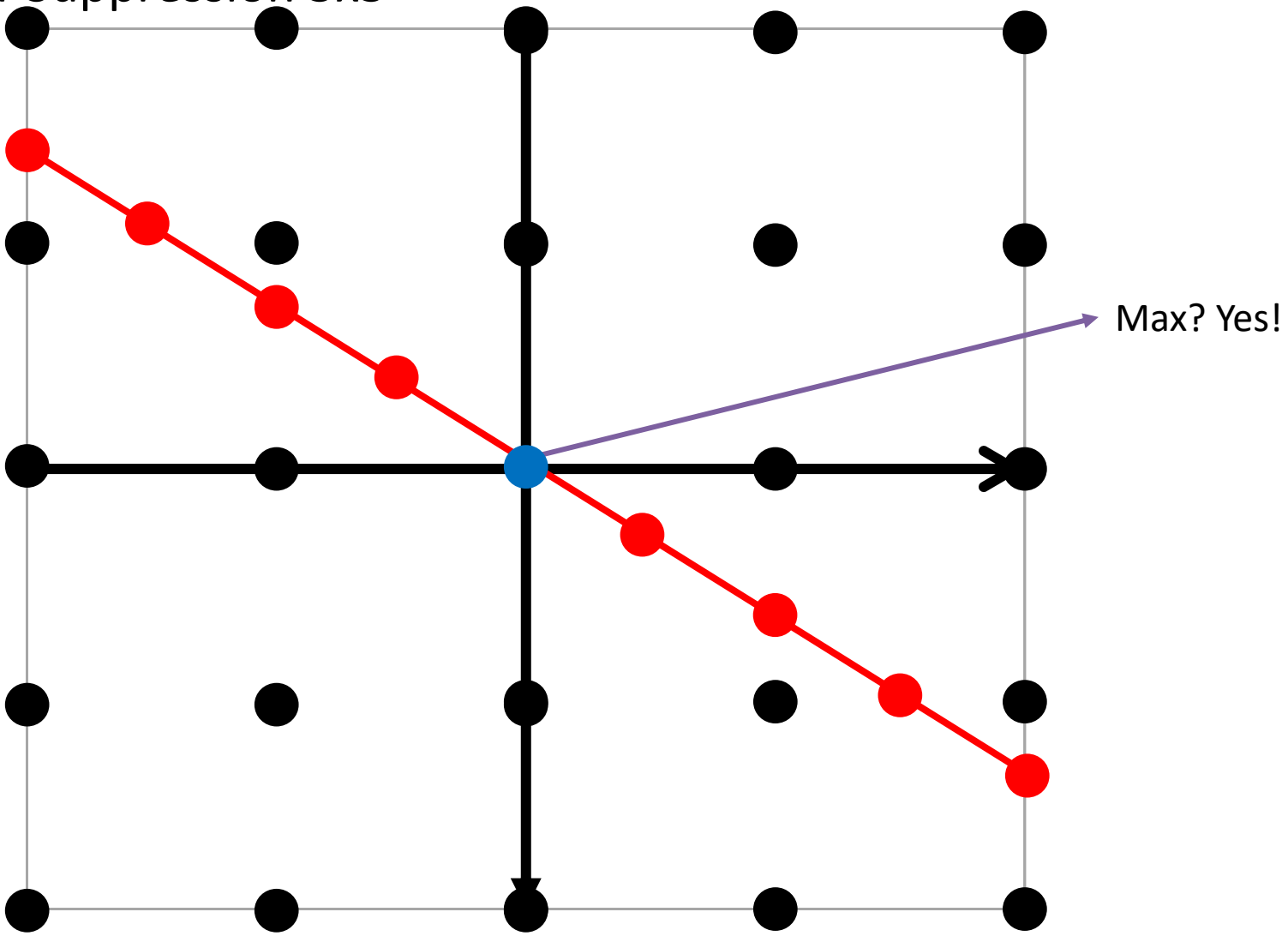
Canny Edge 과제 리뷰

- **Canny Edge 과제 리뷰**
  - Non-Maximum-Suppression 5x5



Canny Edge 과제 리뷰

- **Canny Edge 과제 리뷰**
  - Non-Maximum-Suppression 5x5

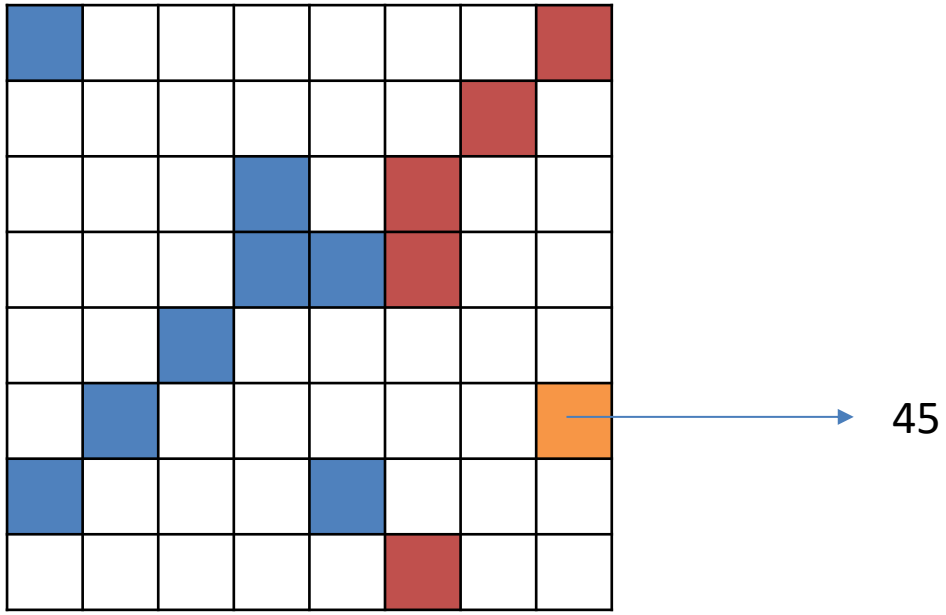


과제

- **Canny Edge 과제 리뷰**
  - Strong edge는 255로 처리

Strong edge

weak edge



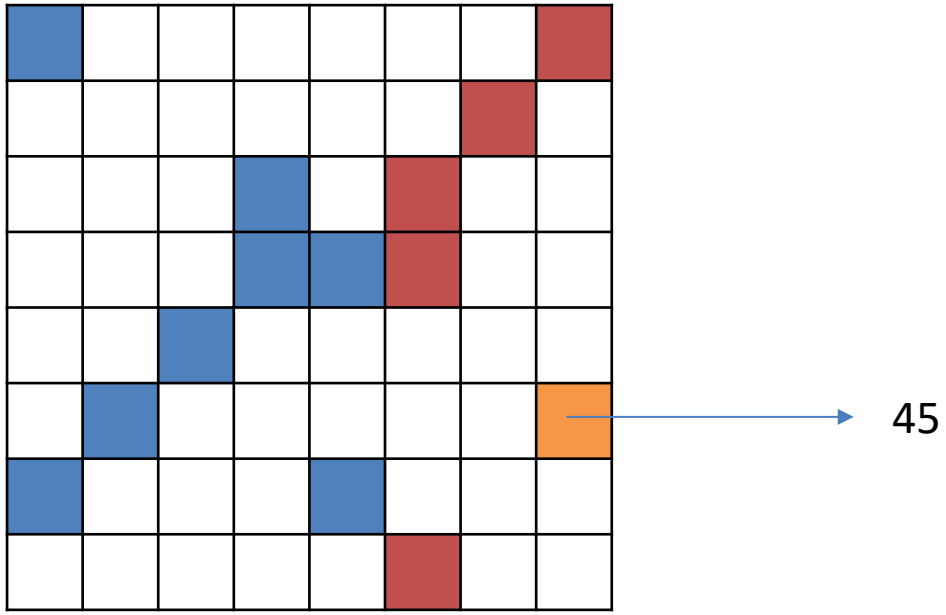
Weak edge list
(0, 0)
(4,2)
…
(3,5)

과제

- **Canny Edge 과제 리뷰**
  - Weak edge는 일단 남김.
  - Weak edge도 아닌 픽셀은 전부 0 처리

Strong edge

weak edge



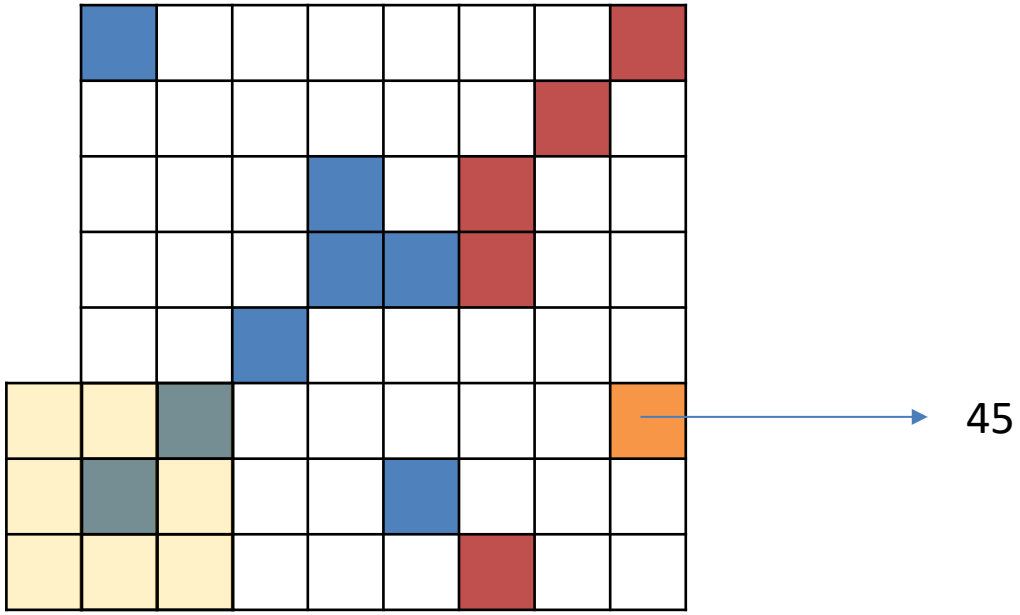
Weak edge list
(0, 0)
(4,2)
…
(3,5)

과제

- **Canny Edge 과제 리뷰**
  - 8 neighborhood search를 통해서 weak edge를 재귀로 탐색

Strong edge

weak edge



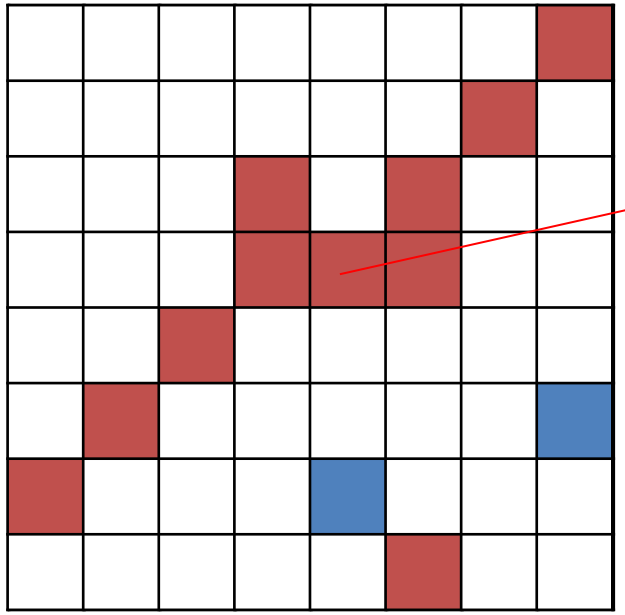
Weak edge list
(0, 0)
(4,2)
…
(3,5)

과제

- **Canny Edge detection**
  - Double threshold
    - High threshold: 60
    - Low threshold: 30

Strong edge

weak edge



Search weak edge
주변 픽셀에 strong edge가 있음

>> 지금까지 연결된 edge는 strong edge가 된다.

공지사항

- **영상처리 02 · 03분반 과제 채점 기준 공개**
  - Canny Edge Detection 채점 기준 공개
    - 기본적으로 02 · 03 분반 코드 모두 동일
    - 코드 기준
      - 총 5가지 TODO 항목에 대하여 평가
      - calcMagnitude, calcAngle
      - non_maximum_suppression_five_size
      - pixel_bilinear_coordinate
      - double_thresholding

공지사항

- **영상처리 02 · 03분반 과제 채점 기준 공개**
  - Canny Edge Detection 채점 기준 공개
    - 코드 기준
      - 총 5가지 TODO 항목에 대하여 평가
        - calcMagnitude, calcAngle – 부분 점수 없이 잘못 구현 시 각 1점 감점

```python
def calcMagnitude(Ix, Iy):
    ##########################################
    # TODO                                   #
    # calcMagnitude 완성                     #
    # magnitude : ix와 iy의 magnitude        #
    ##########################################
    # Ix와 Iy의 magnitude를 계산
    magnitude = None
    return magnitude
```

```python
def calcAngle(Ix, Iy):
    ##########################################
    # TODO                                   #
    # calcAngle 완성                         #
    # angle     : ix와 iy의 angle            #
    ##########################################
    angle = None
    return angle
```

공지사항

- **영상처리 02 · 03분반 과제 채점 기준 공개**
  - Canny Edge Detection 채점 기준 공개
    - 코드 기준
      - 총 5가지 TODO 항목에 대하여 평가
        - non_maximum_suppression_five_size – 각도 표현 잘못 시 1점 감점

```python
def non_maximum_supression_five_size(magnitude, angle, step = 0.5):
    ###############################################################################
    # TODO
    # non_maximum_supression 완성
    # largest_magnitude     : non_maximum_supression 결과(가장 강한 edge만 남김)
    ###############################################################################
    (h, w) = magnitude.shape
    # angle의 범위 : -90 ~ 90
    largest_magnitude = np.zeros((h, w))
    for row in range(2, h-2):
        for col in range(2, w-2):
            degree = angle[row, col]
            # gradient의 degree는 edge와 수직방향이다.
            if 0 <= degree and degree < 45:


            elif 45 <= degree and degree <= 90:


            elif -45 <= degree and degree < 0:


            elif -90 <= degree and degree < -45:


            else:
                print(row, col, 'error!  degree :', degree)

    return largest_magnitude
```

공지사항

- **영상처리 02 · 03분반 과제 채점 기준 공개**
  - Canny Edge Detection 채점 기준 공개
    - 코드 기준
      - 총 5가지 TODO 항목에 대하여 평가
        - non_maximum_suppression_five_size – step별 픽셀 위치 잘못 구현 시 1점 감점

```
def non_maximum_supression_five_size(magnitude, angle, step = 0.5):
    ###############################################################################
    # TODO
    # non_maximum_supression 완성
    # largest_magnitude     : non_maximum_supression 결과(가장 강한 edge만 남김)
    ###############################################################################
    (h, w) = magnitude.shape
    # angle의 범위 : -90 ~ 90
    largest_magnitude = np.zeros((h, w))
    for row in range(2, h-2):
        for col in range(2, w-2):
            degree = angle[row, col]
            # gradient의 degree는 edge와 수직방향이다.
            if 0 <= degree and degree < 45:


            elif 45 <= degree and degree <= 90:


            elif -45 <= degree and degree < 0:


            elif -90 <= degree and degree < -45:


            else:
                print(row, col, 'error!  degree :', degree)

    return largest_magnitude
```

공지사항

- **영상처리 02 · 03분반 과제 채점 기준 공개**
  - Canny Edge Detection 채점 기준 공개
    - 코드 기준
      - 총 5가지 TODO 항목에 대하여 평가
        - non_maximum_suppression_five_size – 비교부분 코드 누락 시 1점 감점

```python
def non_maximum_supression_five_size(magnitude, angle, step = 0.5):
    ###############################################################################
    # TODO
    # non_maximum_supression 완성
    # largest_magnitude     : non_maximum_supression 결과(가장 강한 edge만 남김)
    ###############################################################################
    (h, w) = magnitude.shape
    # angle의 범위 : -90 ~ 90
    largest_magnitude = np.zeros((h, w))
    for row in range(2, h-2):
        for col in range(2, w-2):
            degree = angle[row, col]
            # gradient의 degree는 edge와 수직방향이다.
            if 0 <= degree and degree < 45:


            elif 45 <= degree and degree <= 90:


            elif -45 <= degree and degree < 0:


            elif -90 <= degree and degree < -45:


            else:
                print(row, col, 'error!  degree :', degree)

    return largest_magnitude
```

공지사항

- **영상처리 02 · 03분반 과제 채점 기준 공개**
  - Canny Edge Detection 채점 기준 공개
    - 코드 기준
      - 총 5가지 TODO 항목에 대하여 평가
        - Double thresholding – 잘못 구현 시 2점 감점
        - 그러나 thresholding 과정 중 high, low 등호 생략은 미감점

```
def double_thresholding(src, high_threshold):

    ###############################################################################
    # TODO
    # double_thresholding 완성
    # Goal : Weak Edge와 Strong Edge를 토대로 연결성을 찾아서 최종적인 Canny Edge Detection 이미지를 도출
    # 이 함수는 건드릴 필요가 없음.
    # largest_magnitude     : non_maximum_supression 결과(가장 강한 edge만 남김)
    # double_thresholding 수행 high threshold value는 메인문에서 지정한 값만 사용할 것
    # three : 40
    # five : 29
    ###############################################################################

    dst = src.copy()
    # dst => 0 ~ 255
    dst -= dst.min()
    dst /= dst.max()
    dst *= 255
    dst = dst.astype(np.uint8)
    (h, w) = dst.shape

    high_threshold_value = high_threshold

    low_threshold_value = high_threshold_value * 0.4
```

공지사항

- **영상처리 02 · 03분반 과제 채점 기준 공개**
  - Canny Edge Detection 채점 기준 공개
    - 보고서 기준
    - 제출 이미지 누락 시 1점 감점
    - 아래 내용이 없을 시 1점 감점

- **Canny edge detection**
  - 보고서 필수 내용

  - 3x3 Canny Edge Detection 수행 시 Threshold 값이랑 5x5 Canny Edge Detection 수행 시 Threshold 값이랑 비교했을때 왜 5X5가 더 낮은지 분석 내용 추가.

실습

- **Salt and Pepper Noise**
  - 이미지에 랜덤하게 흰색(salt)과 검은색(pepper) 픽셀을 추가하는 방식

Original

Original + Noise

실습

- **Salt and Pepper Noise**
  - Average filter와 Median filter를 사용하여 Noise 제거

Original + Noise

Average Filter

Median Filter

실습

- **Salt and Pepper Noise**
  - Salt and Pepper Noise의 경우에는 **Median Filter가 효과적**

Original + Noise

Average Filter

Median Filter

실습

- **Salt and Pepper Noise**
  - Median Filter는 mask 영역내의 이미지 값들을 정렬 후 중앙값을 취함

50	65	52
63	255	58
61	60	57

→ 50 52 57 58 **60** 61 63 65 255 → 60

```
def my_median_filtering(src,msize):
    h,w = src.shape
    dst = np.zeros((h,w))
    for row in range(h):
        for col in range(w):
            # (row,col)를 중심으로 mask를 씌웠을때 index를 초과하는 영역이 생겨남
            # 이 index를 초과하는 범위에 대해서는 median filter 적용시 해당 사항이 없도록 하기 위해서
            # 다음과 같은 row,col를 조정
            r_start = np.clip(row - (msize // 2), 0, h)
            r_end = np.clip(row + (msize // 2), 0, h)

            c_start = np.clip(col-msize // 2, 0, h)
            c_end = np.clip(col+msize // 2, 0, h)
            mask = src[r_start:r_end, c_start:c_end]
            dst[row,col] = np.median(mask)

    return dst.astype(np.uint8)
```

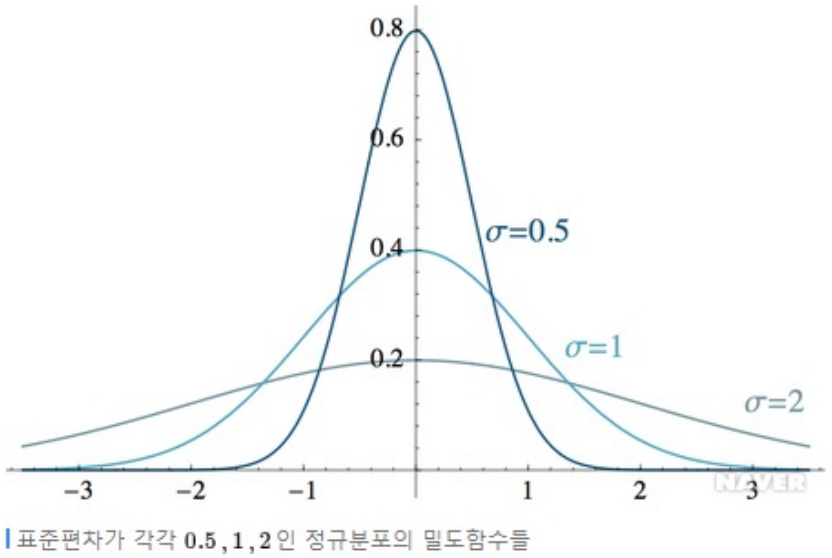
Median Filter

실습

- **Gaussian Noise**
  - Normal Gaussian 분포에서 값을 랜덤하게 샘플링한 값을 Noise로 사용
  - 기본적으로 평균이 0이고 표준편차가 1이라고 가정(즉 Noise 값은 0 ~ 1)

```python
def add_gaus_noise(src, mean=0, sigma=0.1):

    """
    :param src: gaussian noise를 적용할 이미지 (0 ~ 255) 사이의 값을 가짐
    :param mean: 평균
    :param sigma: 표준편차
    :return: noise가 추가된 이미지
    """

    dst = src / 255
    (h, w) = dst.shape
    noise = np.random.normal(mean, sigma, size=(h, w))
    dst += noise

    return my_normalize(dst)
```

$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$



표준편차가 각각 0.5, 1, 2인 정규분포의 밀도함수들

실습

- **Gaussian Noise**
  - Noise는 Noise가 없은 이미지($I(x,y)$)에 더하여 생성

```python
def add_gaus_noise(src, mean=0, sigma=0.1):

    """
    :param src: gaussian noise를 적용할 이미지 (0 ~ 255) 사이의 값을 가짐
    :param mean: 평균
    :param sigma: 표준편차
    :return: noise가 추가된 이미지
    """

    dst = src / 255
    (h, w) = dst.shape
    noise = np.random.normal(mean, sigma, size=(h, w))
    dst += noise

    return my_normalize(dst)
```

$$I_G(x,y) = I(x,y) + N(x,y)$$

실습

- **Gaussian Noise**
  - 결과 이미지

Original

Original + Noise

실습

- **Gaussian Noise Removal**
  - Average Filtering
    - 필터의 크기에 따라 Noise가 제거되는 정도가 다르다

5 × 5

7 × 7

9 × 9

실습

- **Gaussian Noise Removal**
  - Gaussian Filtering
    - 필터의 크기와 시그마에 크기에 따라 Noise가 제거되는 정도가 다르다
    - 시그마 값을 3으로 고정하고 필터의 크기를 조절

5×5

7×7

9×9

실습

- **Gaussian Noise Removal**
  - Gaussian Filtering
    - 필터의 크기와 시그마에 크기에 따라 Noise가 제거되는 정도가 다르다
    - 필터의 크기를 7로 고정하고 시그마 값을 조절

$\sigma = 0.1$

$\sigma = 1$

$\sigma = 5$

실습

- **Gaussian Noise Removal**
  - Average N noise images
    - 평균이 0인 Gaussian 분포에서 랜덤하게 샘플링한 값을 Noise로 사용
    - 이때, Noise를 같은 이미지에 N개를 생성 후 이미지들을 평균

$$\frac{1}{N}\sum_{i=1}^{N} I_G^i\,(x,y) = I(x,y) + \frac{1}{N}\sum_{i=1}^{N} N^i\,(x,y)$$

Average N noise images

과제

- **Bilateral Filtering**
  - Bilateral Filtering 구현을 목적으로 함.

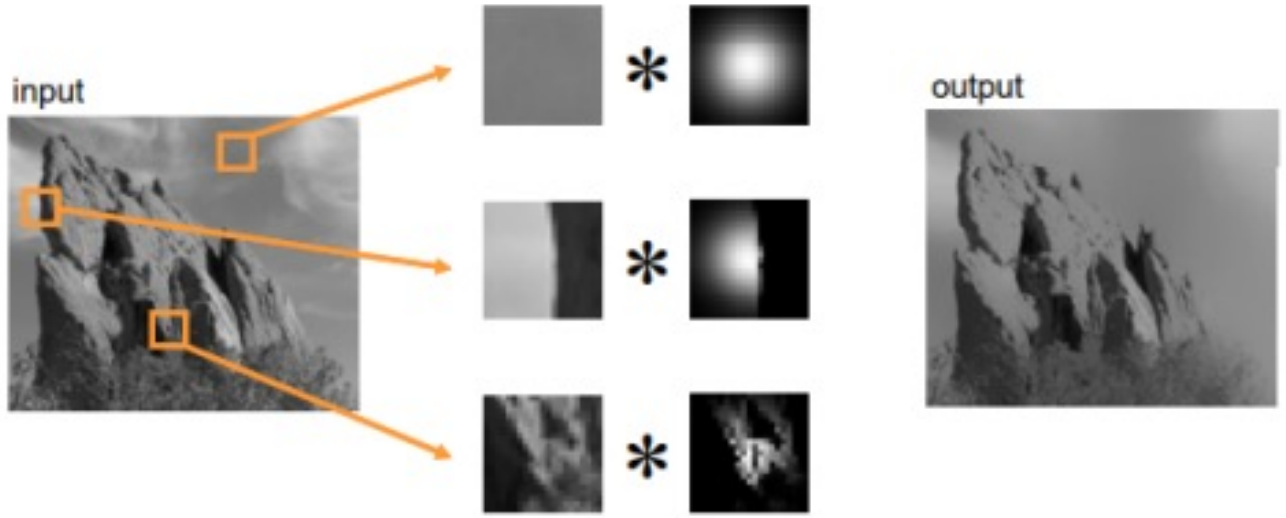
Gaussian Noise

Bilateral Filtering

과제

- **Bilateral Filtering**
  - Bilateral 수식

$$f(i,j,k,l) = \frac{1}{\mathcal{N}} \exp\left(-\frac{(i-k)^2}{2\sigma_x^2} - \frac{(j-l)^2}{2\sigma_y^2}\right) \cdot \exp\left(-\frac{\left(I(i,j) - I(k,l)\right)^2}{2\sigma_r^2}\right)$$



The kernel shape depends on the image content.

과제

- **Bilateral Filtering**
  - Bilateral Filtering 수식에 추가적으로 DoG 결과를 적용한 이미지도 반영하도록함

$$f(i,j,k,l) = \frac{1}{N}\exp\left(-\frac{(i-k)^2}{2\sigma_x^2} - \frac{(j-l)^2}{2\sigma_y^2}\right)\cdot\exp\left(-\frac{(I(i,j)-I(k,l))^2}{2\sigma_r^2}\right)\cdot\exp\left(-\frac{(\mathrm{DoG}(i,j)-\mathrm{DoG}(k,l))^2}{2\sigma_d^2}\right)$$

과제

- **Bilateral Filtering**
  - Bilateral Filtering 구현 부분

```
(h, w) = src.shape
# filter size만큼의 y,x 좌표 생성
y, x = np.mgrid[-(msize // 2): (msize // 2) + 1, -(msize // 2): (msize // 2) + 1]
# filte size만큼 padding 이미지 생성
img_pad = my_padding(src, (msize // 2, msize // 2), 'zero')
# padding 폭
(p_h, p_w) = (msize // 2, msize // 2)

dog_1_y, dog_1_x = get_DoG_filter_by_expression(5, sigma_dog)
dog_y_image = cv2.filter2D(src.astype(np.float32), -1, dog_1_y, borderType=cv2.BORDER_CONSTANT)
dog_x_image = cv2.filter2D(src.astype(np.float32), -1, dog_1_x, borderType=cv2.BORDER_CONSTANT)

dog_img = np.sqrt(dog_x_image ** 2 + dog_y_image ** 2)
dog_img = dog_img / dog_img.max()

dog_pad = my_padding(dog_img, pad_shape=(p_h, p_w))
# dst
dst = np.zeros((h, w), dtype=np.float32)
```

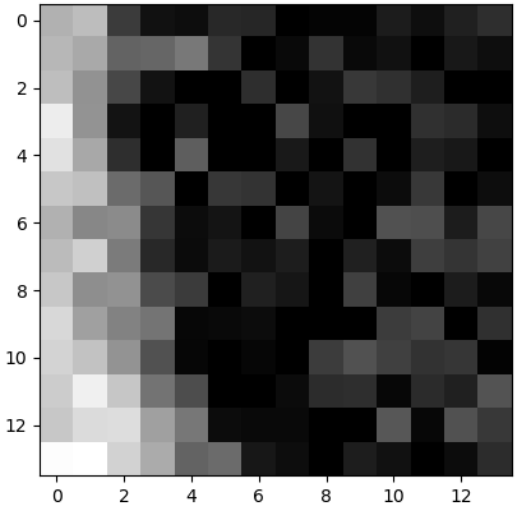
과제

- **Bilateral Filtering**
  - mask 변수에 해당하는 부분을 구현

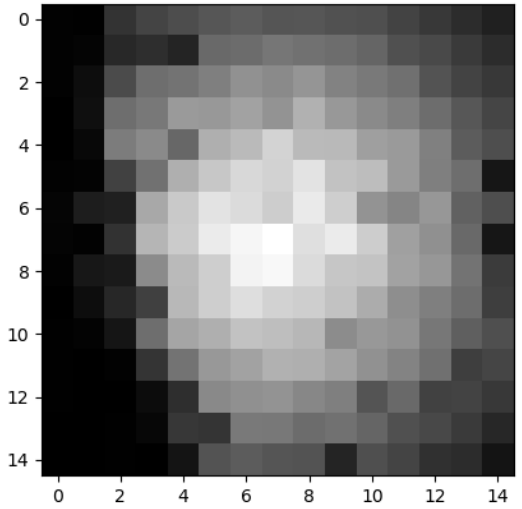
```
for i in range(h):
    print('\r%d / %d ...' %(i, h), end="")
    for j in range(w):

        mask = ???

        if i == pos_x and j == pos_y:
            import matplotlib.pyplot as plt
            mask_visual = mask
            mask_visual = mask_visual - mask_visual.min()
            mask_visual = (mask_visual / mask_visual.max() * 255).astype(np.uint8)
            cv2.imshow('mask', mask_visual)
            img = img_pad[i:i + msize - 1, j:j + msize - 1]
            img = my_normalize(img)
            plt.imshow(img, cmap='gray')
            plt.show()
            plt.imshow(mask_visual, cmap='gray')
            plt.show()
        # 한 픽셀에 대한 mask filtering 결과를 반영한다.
        dst[i, j] = np.sum(img_pad[i: i + msize, j: j + msize] * mask)
```

과제

- **Bilateral Filtering**
  - pos_x, pos_y 해당하는 패치 시각화



Gaussian Noise



Bilateral Filtering

과제

- **Bilateral Filtering**
  - Main 함수 부분에서 filter size, sigma, sigma_s, sigma_r 파라미터 값을 찾아서 완성.

```
#######################################################
# TODO
# my_bilateral, gaussian mask 채우기
# filter size, sigma, sigma_s, sigma_r 값 채우기
# DoG를 활용하는 부분도 채우기
#######################################################
dst = my_bilateral(src_noise, ??, sigma_s=??, sigma_r=??, sigma_d=0.08, sigma_dog=3, pos_x=pos_x, pos_y=pos_y)
dst = my_normalize(dst)

dog_dst = my_bilateral(src_noise, ??, sigma_s=??, sigma_r=??, sigma_d=0.08, sigma_dog=3,pos_x=pos_x, pos_y=pos_y,
                       pad_type='zero', is_dog=True)
dog_dst = my_normalize(dog_dst)
```

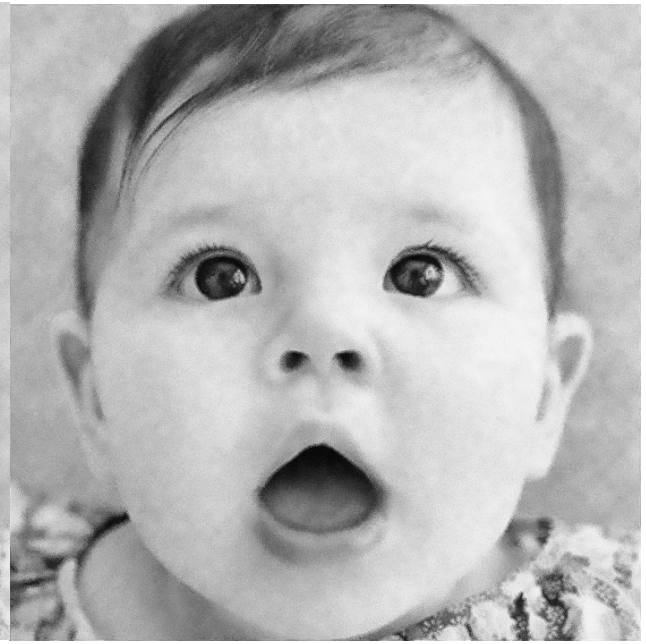
과제

- **Bilateral Filtering**
  - 결과 이미지

Gaussian Noise

Gaussian Filtering

Bilateral Filtering



Bilateral Filtering + DoG

과제

- **Bilateral Filtering**
  - 결과 이미지

Bilateral Filtering + DoG – Bilateral Filtering

과제

- **Bilateral Filtering**
  - 보고서 사진 첨부 목록
    - 결과이미지 5장 첨부 Page 39페이지 4장 Page 40페이지 1장
  - 과제 진행 2023.5월 19일 ~ 2023년 6월 2일 23:59

Q & A