고려대학교
KOREA UNIVERSITY

Pixel 히스토그램 / 기타 학습 내용

PARALELL ALGORITHM LAB

Mail

고려대학교

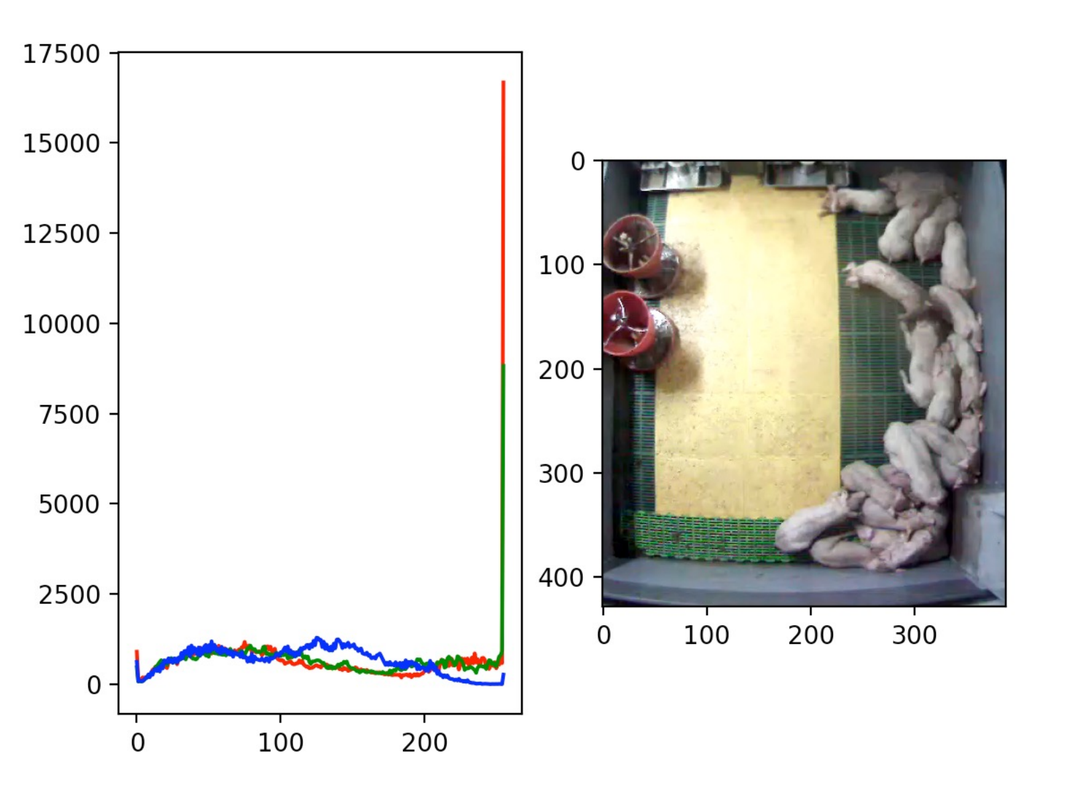
목차

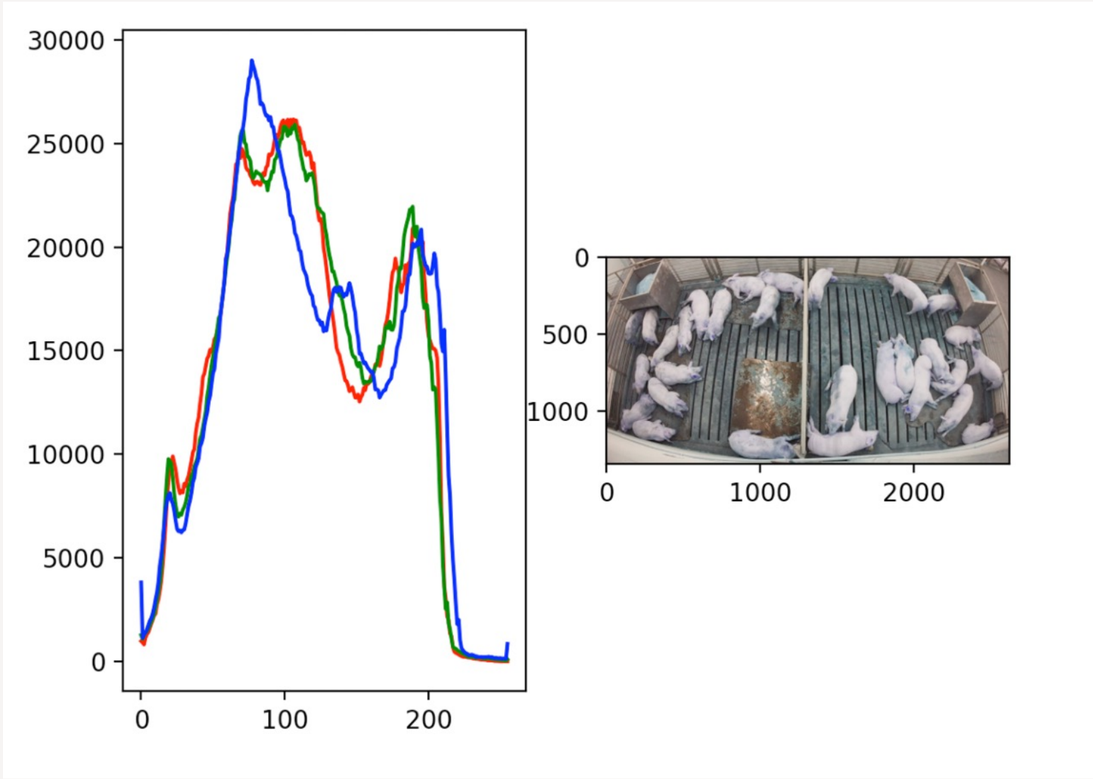
- Pixel 정보 히스토그램 그리기 (Color)
- Pxiel 정보 히스토그램 그리기 (Gray)
- Sharpening & 평탄화
- 7bit gray 변환
- Pooling

고려대학교

▪ Pixel 정보 히스토그램 그리기 (Color)

RGB 채널 분리 (컬러)

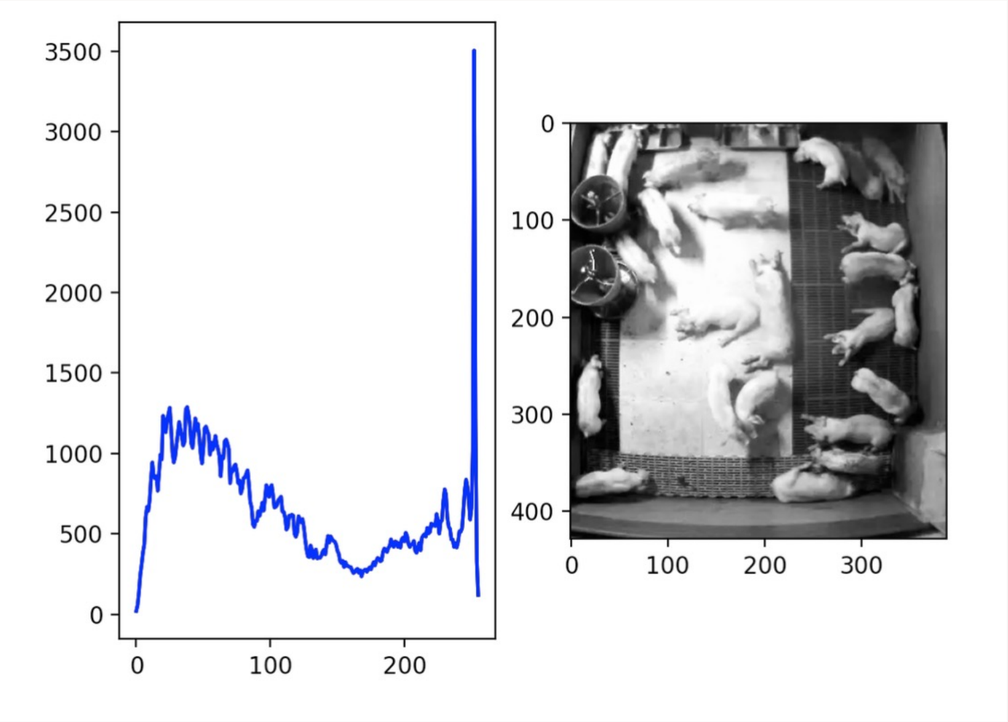




▪ Pxiel 정보 히스토그램 그리기 (Gray)

RGB 채널 분리 (흑백)

이전에 실험했던 것과 달리 휴대폰으로 흑백 변환한 영상을 RGB 분리했을 때는 채널 별로 동일한 Pixel 값을 가지는 것을 확인함
Gray 변환 시에 사용되는 method에 따라 약간의 차이가 있는 것으로 보임



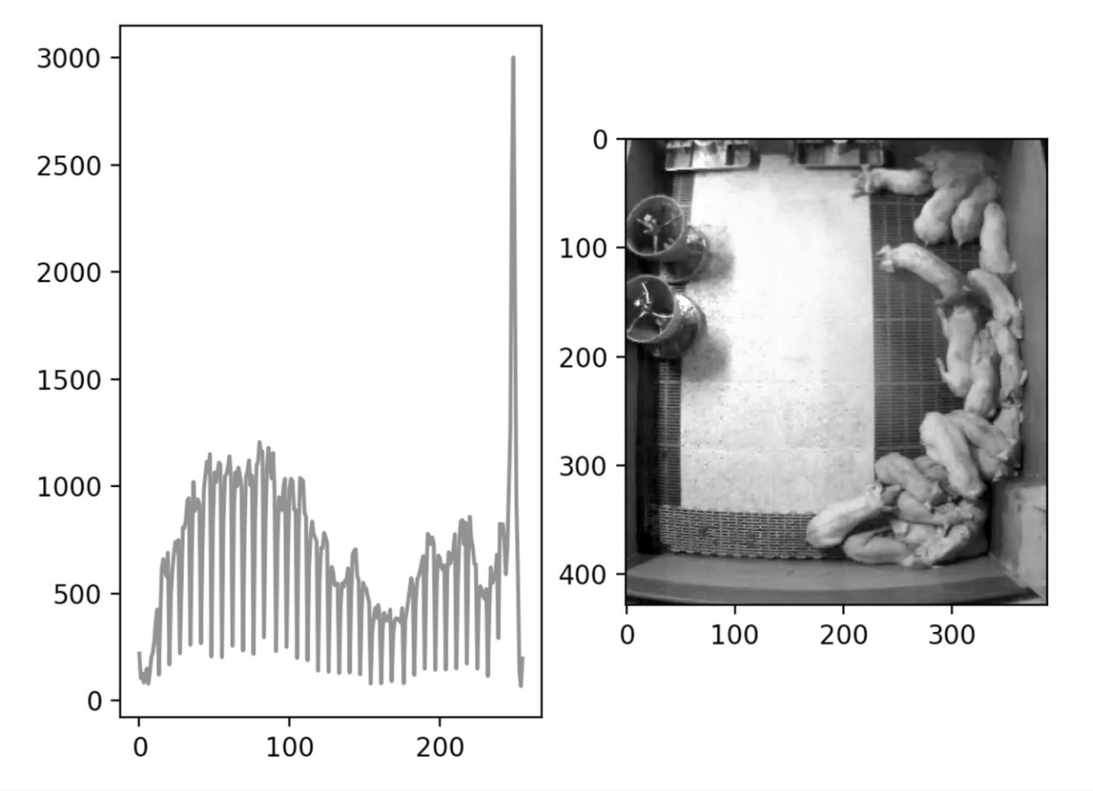

```
hist_R [[  20.]
 [  57.]
 [ 137.]
 [ 245.]
 [ 310.]
 [ 385.]
 [ 435.]
 [ 602.]
 [ 668.]
 [ 643.]
 [ 710.]
 [ 845.]
 [ 944.]
 [ 855.]
 [ 845.]
 [ 858.]
 [ 768.]
 [ 883.]
 [ 992.]
 [ 962.]
 [1234.]
 [1202.]
 [1132.]
 [1208.]
 [1247.]
 [1284.]
 [1154.]
 [1004.]
 [ 944.]
 [ 979.]
 [1052.]
 [1126.]
 [1196.]
 [1147.]
 [1102.]
 [1049.]
 [1071.]
 [1275.]
 [1288.]
 [1240.]
 [1182.]
 [1062.]
 [1034.]
 [1121.]
```

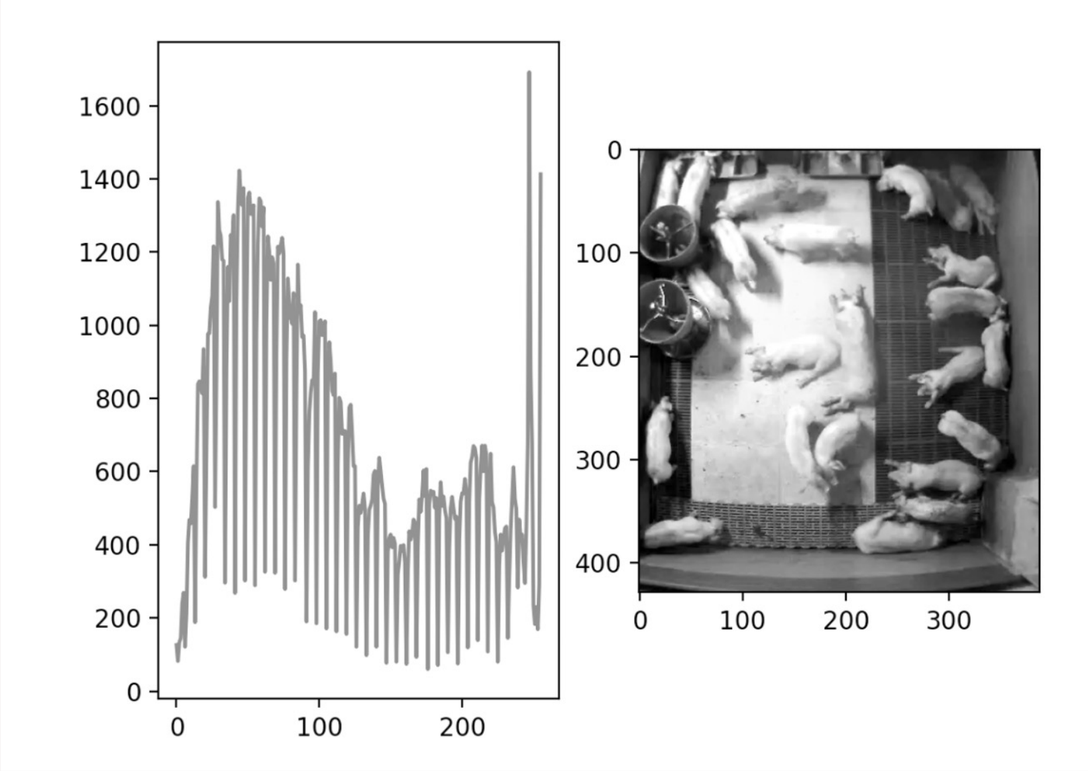
```
hist_G [[  20.]
 [  57.]
 [ 137.]
 [ 245.]
 [ 310.]
 [ 385.]
 [ 435.]
 [ 602.]
 [ 668.]
 [ 643.]
 [ 710.]
 [ 845.]
 [ 944.]
 [ 855.]
 [ 845.]
 [ 858.]
 [ 768.]
 [ 883.]
 [ 992.]
 [ 962.]
 [1234.]
 [1202.]
 [1132.]
 [1208.]
 [1247.]
 [1284.]
 [1154.]
 [1004.]
 [ 944.]
 [ 979.]
 [1052.]
 [1126.]
 [1196.]
 [1147.]
 [1102.]
 [1049.]
 [1071.]
 [1275.]
 [1288.]
 [1240.]
 [1182.]
 [1062.]
 [1034.]
 [1121.]
```

```
hist_B [[  20.]
 [  57.]
 [ 137.]
 [ 245.]
 [ 310.]
 [ 385.]
 [ 435.]
 [ 602.]
 [ 668.]
 [ 643.]
 [ 710.]
 [ 845.]
 [ 944.]
 [ 855.]
 [ 845.]
 [ 858.]
 [ 768.]
 [ 883.]
 [ 992.]
 [ 962.]
 [1234.]
 [1202.]
 [1132.]
 [1208.]
 [1247.]
 [1284.]
 [1154.]
 [1004.]
 [ 944.]
 [ 979.]
 [1052.]
 [1126.]
 [1196.]
 [1147.]
 [1102.]
 [1049.]
 [1071.]
 [1275.]
 [1288.]
 [1240.]
 [1182.]
 [1062.]
 [1034.]
 [1121.]
```

▪ Pxiel 정보 히스토그램 그리기 (Gray)

Gray scale

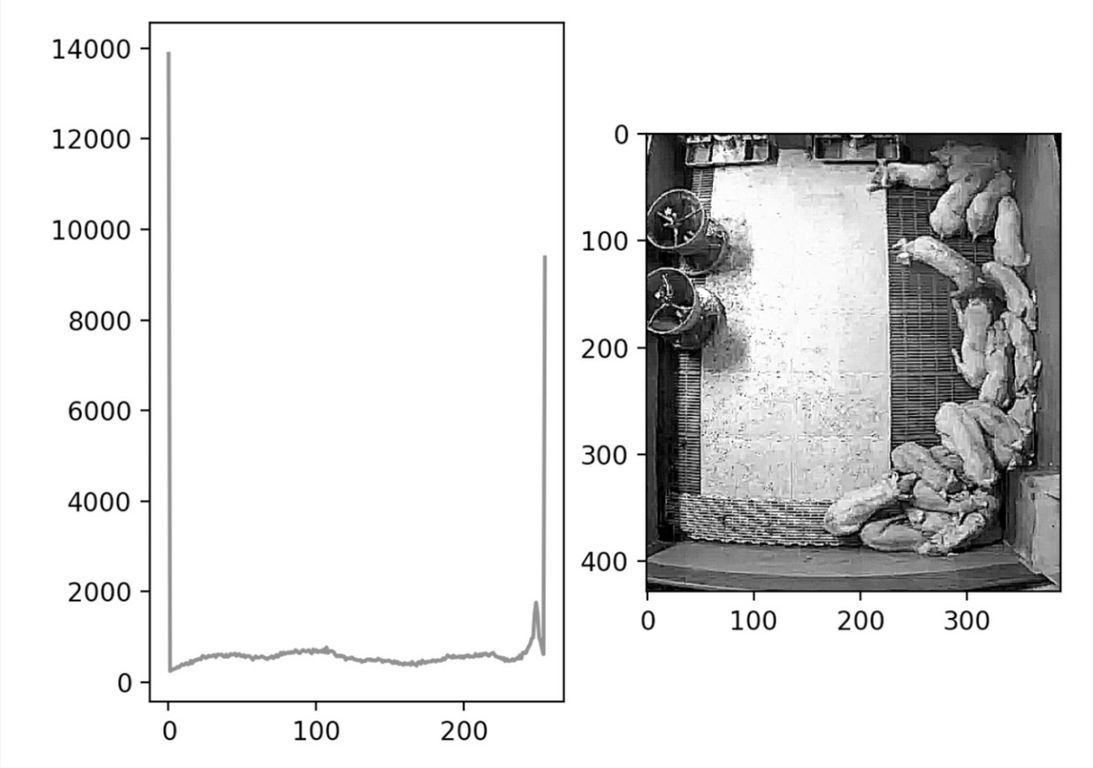




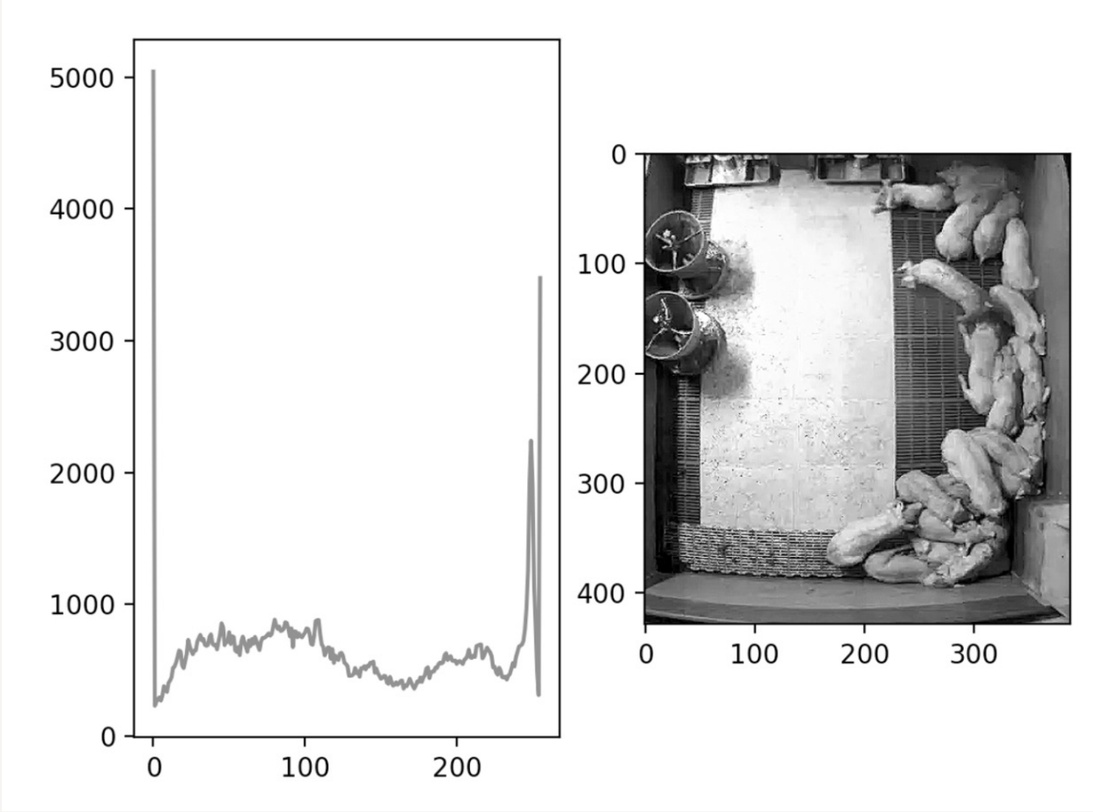
▪ Sharpening & 평탄화

sharpening filter

1. [[-1, -1, -1], [-1, 9, -1], [-1, -1, -1]]



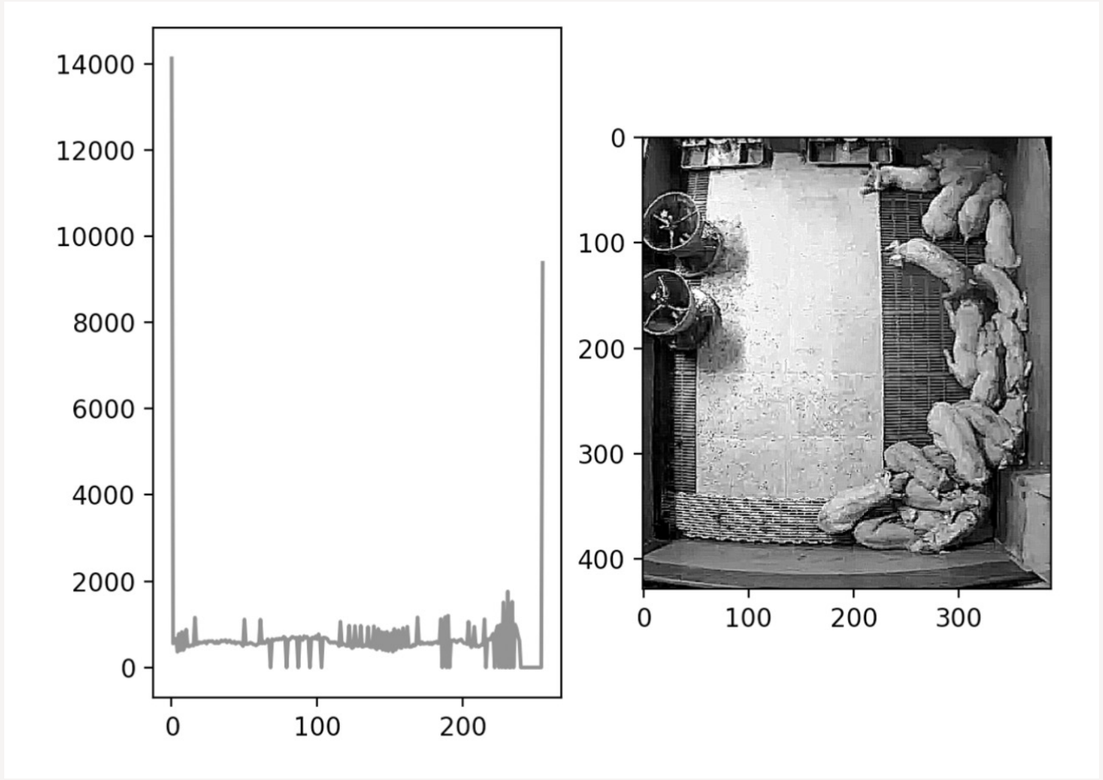
2. [[0, -1, 0], [-1, 5, -1], [0, -1, 0]]



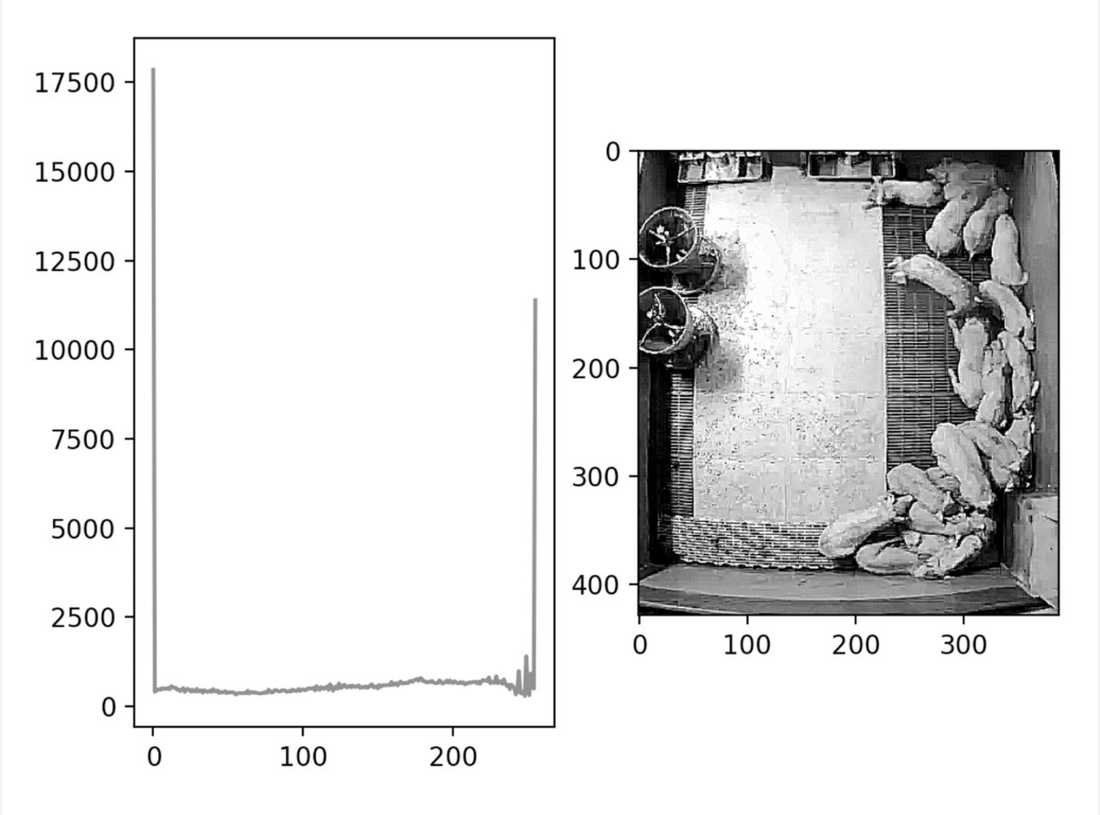
▪ Sharpening & 평탄화

sharpening filter & cv2.equalizeHist

1. sharpening → equalizeHist



2. equalizeHist → sharpening



▪ Sharpening & 평탄화

sharpening filter & CLAHE

평탄화 적용 시 너무 밝아져 경계선을 알아보기 어려운 경우 많음 → **이미지를 일정한 영역(아래 코드에서 tileGridSize 파라미터)으로 나누어 평탄화를 적용**

그렇게 해도 일정한 영역 내에서 극단적으로 어둡거나 밝은 부분이 있으면 노이즈가 생겨 원하는 결과를 얻을 수 없음 → **어떤 영역이든 지정된 제한 값(아래 코드에서 clipLimit 파라미터)을 넘으면 그 픽셀은 다른 영역에 균일하게 배분하여 적용하는 평탄화 방식**

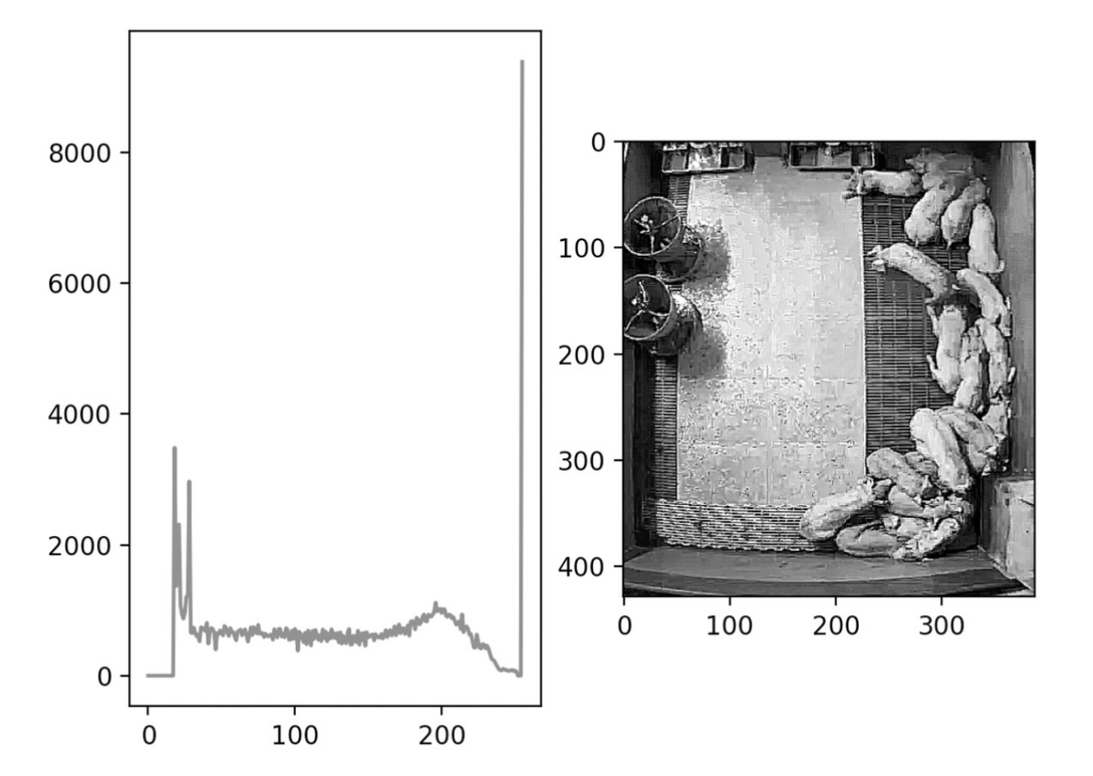
and time) simultaneously. Therefore, we apply CLAHE twice with different parameter values in order to maximize the inter-class variation (block size = 2×2, clip limit = 160 and denoted as CLAHE1) and

```
clahe = cv2.createCLAHE(clipLimit=160, tileGridSize=(2,2))
clahe_img = clahe.apply(image)
```

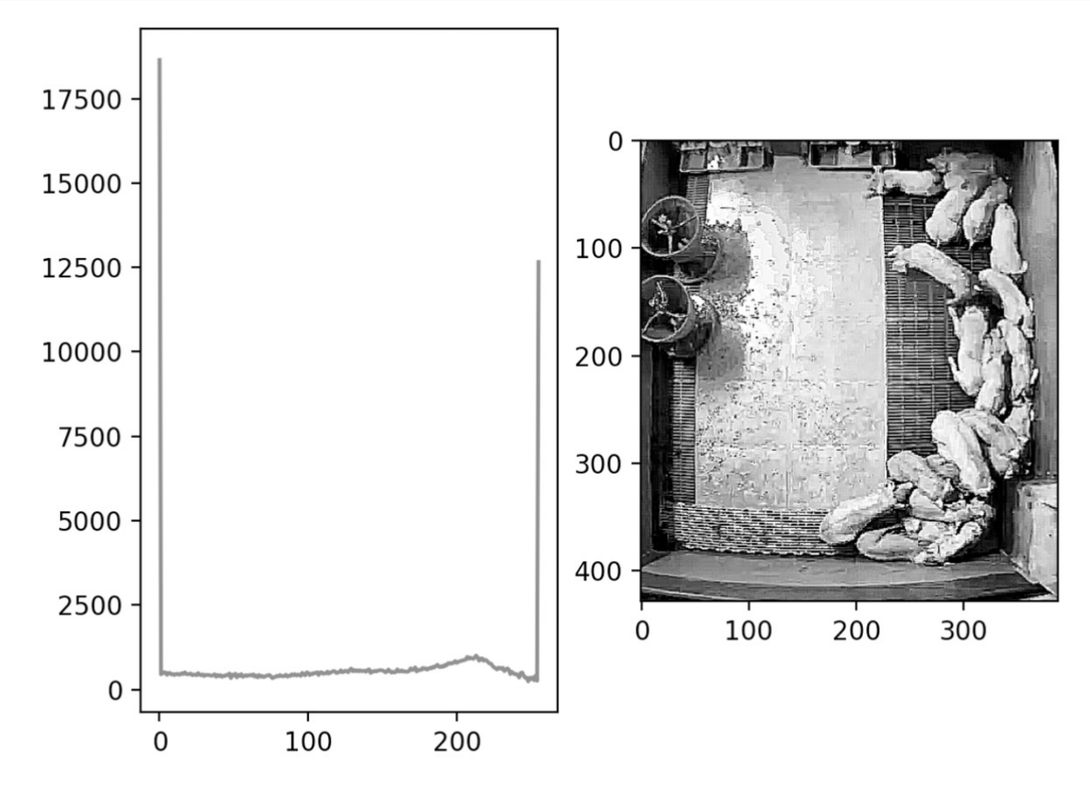
▪ Sharpening & 평탄화

sharpening filter & CLAHE

1. sharpening → CLAHE1

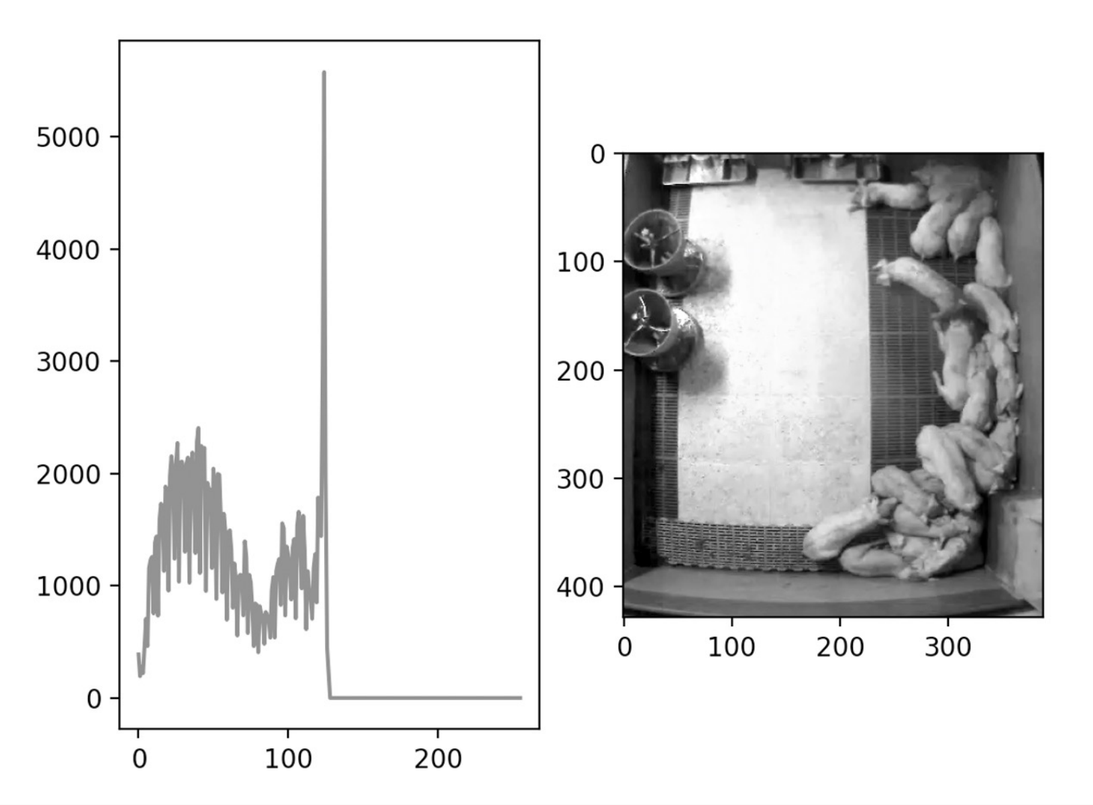


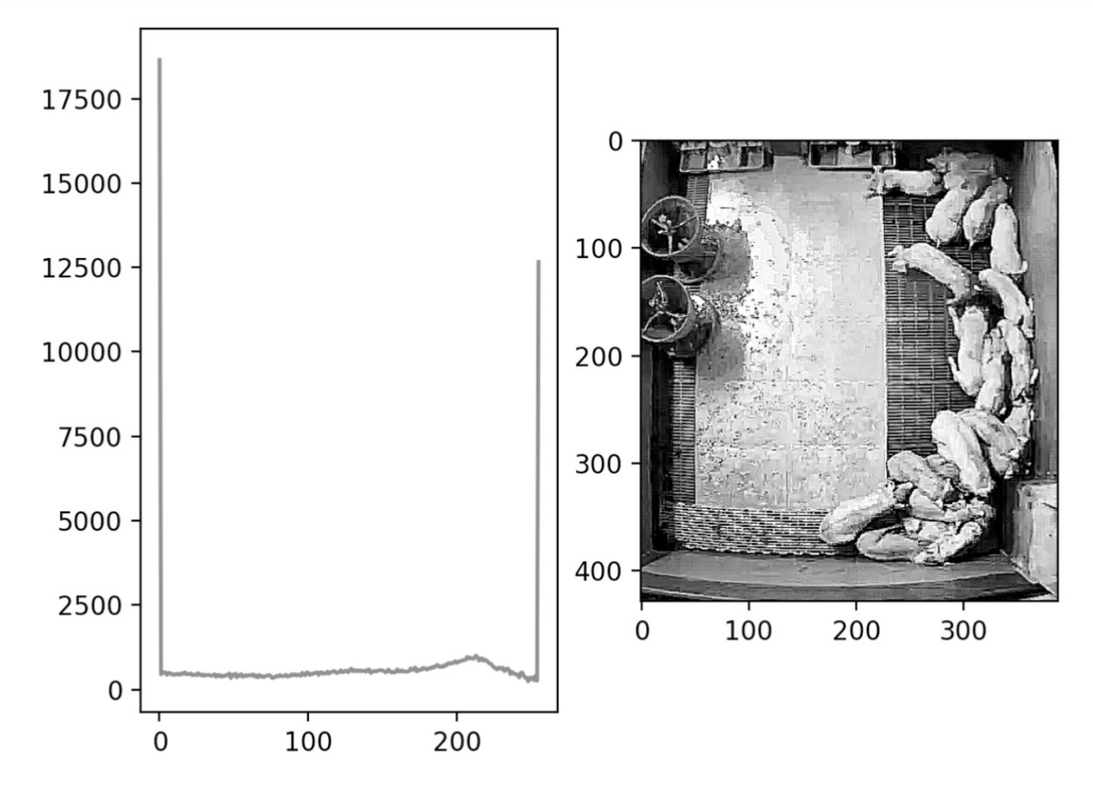
2. CLAHE1 → sharpening



▪ 7bit gray 변환

단순 7비트 변환 (shift 연산)





고려대학교

▪ 7bit gray 변환

전달 받은 코드 분석 (proposed method)

```
def linear_map(x, in_min, in_max, out_min, out_max):
    return int((x - in_min) * (out_max - out_min) / (in_max - in_min) + out_min)
```

최소값과의 입력 값의 차이 * (출력 값의 max-min / 입력 값의 max-min) + 출력 최소값 **-> 원하는 출력 범위로 선형 변환**

```
def log_transform(pixel_value):
    return np.uint8(255 * (np.log(1 + pixel_value) / np.log(1 + 255)))
```

log_transform 함수는 로그 변환을 사용하여 이미지의 대비를 개선하기 위해 사용됨, 기본 식 S = 상수c * log(1 + 입력값)
-> 상수 c = 255 / log(1 + 255) -> 0~255 조정

```
rows, cols = img.shape
for i in range(rows):
    for j in range(cols):
        img[i, j] = log_transform(img[i, j])

min_pixel = np.min(img)
max_pixel = np.max(img)

input_min = min_pixel
input_max = max_pixel
output_min = 64
output_max = 191

for i in range(rows):
    for j in range(cols):
        pixel_value = img[i, j]
        img[i, j] = np.clip(linear_map(pixel_value, input_min, input_max, output_min, output_max), output_min, output_max)
```

-> 전체 픽셀 순회하며
로그 변환 후 선형 변환

고려대학교

▪ 7bit gray 변환

전달 받은 코드 분석 (simple method)

```
img = (img >> 1).astype(np.uint8) + 64
```

Bit shift로 7bit 변환(0 ~ 127) 후 int로 변환, 64 ~ 191의 범위를 맞추기 위해 +64

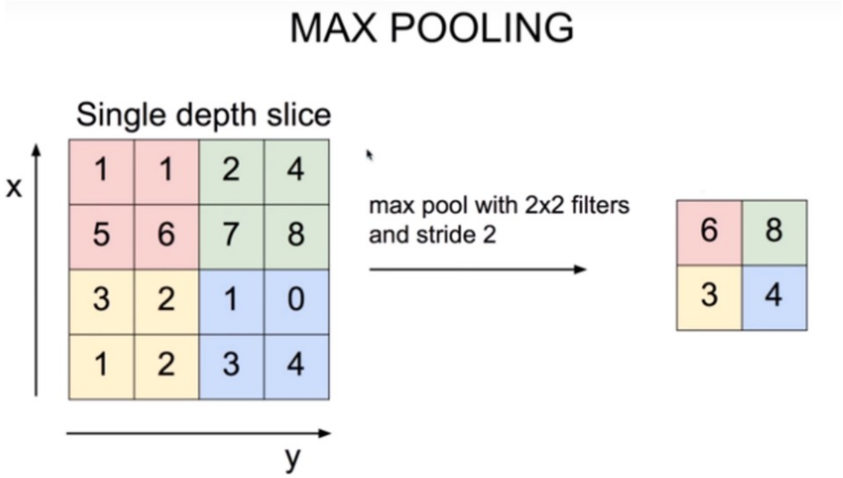
기타 학습 내용 (기초 용어)

Pooling

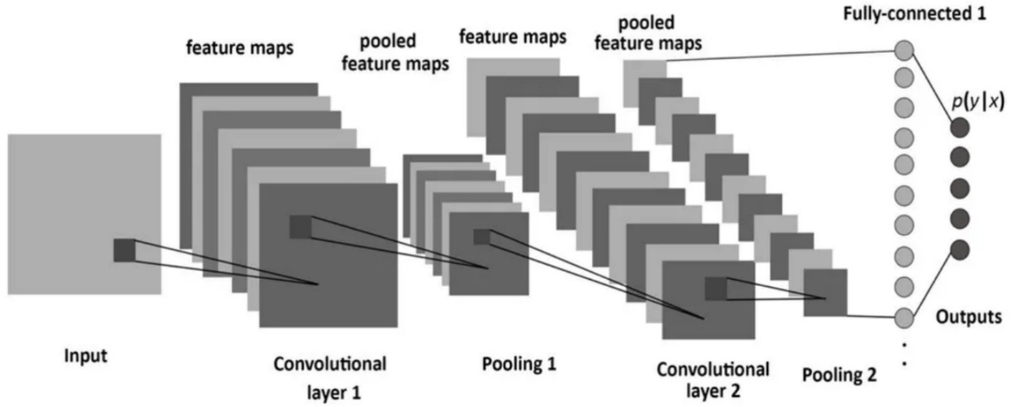
Pooling은 `sub sampling` (Image data를 작은 size의 image로 줄이는 과정)이라고도 함

CNN기준으로는 CONV layer와 Activation을 거쳐 나온 output인 activation feature map에 대하여 적용

- Max pooling (CNN에서 주로 사용, 해당 receptive field에서 가장 큰 값만을 고름)



- Average pooling (receptive field안에 존재하는 parameter 평균값만)
- Stochastic pooling
- Cross channel pooling
- ...



Pooling의 효과

1. parameter를 줄이고 새로운 map을 얻기 때문에, 해당 network의 표현력이 줄어들어 **Overfitting을 억제**

2. computation이 줄어들어 hardware resource(energy)를 절약하고 speedup

Pooling의 특징

1. training을 통해 train되어야 할 parameter가 없다.

2. Pooling의 결과는 channel 수에는 영향이 없으므로 channel 수는 유지된다. (independent)

3. input feature map에 변화(shift)가 있어도 pooling의 결과는 변화가 적다. (robustness)