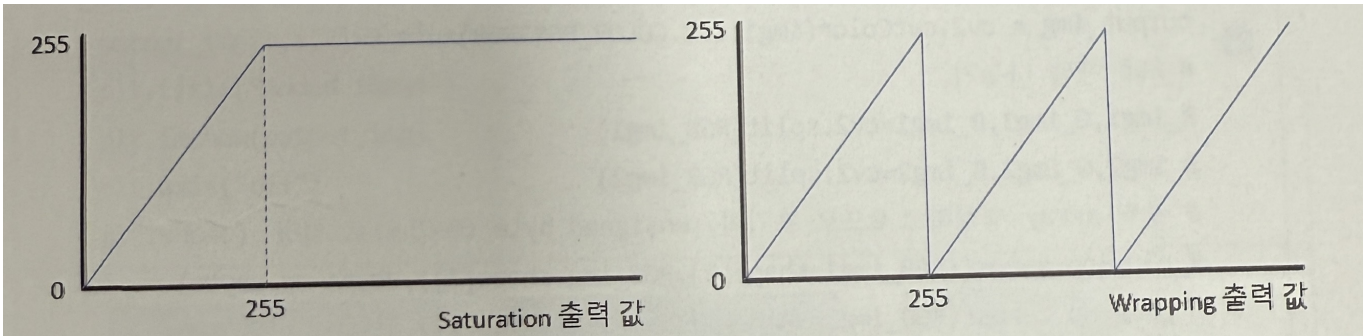
문제1. 영상의 산술 및 논리 연산을 통해 발생할 수 있는 문제점을 기술하고 이를 해결하기 위해 어떠한 방법을 사용하는지 그림을 그리고 서술하여라

- 오버플로우(overflow)
- 'Saturation'은 픽셀값이 설정된 최대값을 초과할 경우 초과된 값을 미리 설정된 최대값(255)으로 교체하는 방법
- 'Wrapping'은 픽셀값이 설정된 최대값을 초과할 경우 초과된 값에서 최대값 + 1을 뺀 값으로 픽셀 값을 교체하는 방법
- '최대-최소 정규화 방법'은 픽셀 I(x, y)을 미리 설정된 최대값과 최소값 사이의 값으로 재계산하는 방법



(그림 3–2) Saturation(왼쪽)과 Wrapping(오른쪽)

문제2. "작은 블록은 대략적으로 균등한 조명을 가질 가능성이 크기 때문에 임계값 적용에 보다 적합"하다는 가정을 기본전제로 하고 있는 기법이 어떤기법인지 서술하고, 해당 기법의 주의할점을 상세히 서술하세요

- 적응적 임계값 적용 방법. 영상의 특성과 크기에 따라 블록의 크기를 가변적으로 결정해 주어야 한다.

문제3. 픽셀 나누기 연산은 동일한 크기의 두 영상으로부터 동일한 좌표의 각 픽셀들을 나눈 값을 결과 영상에 입력하는 방법이다. 연산 된 값이 부동소수점 값이 될 수 있는데 출력 영상에서 어떻게 입력 되어야하고 결과 값은 어떻게 취해야 하는지 서술하시오.

- 출력 영상에서는 정수형으로 입력 되어야한다
- 결과값에서 소숫점 이하를 버림 혹은 반올림을 취한다

문제4. 필터 커널의 크기를 변경할 때, 영상 품질에 어떤 영향을 미치는지 설명하시오. 커널 크기를 작게 설정했을 때와 크게 설정했을 때 각각의 장단점을 서술하고, 그에 맞는 활용 사례를 제시하시오.

- 커널의 크기에 따라 영상의 스무딩의 정도가 결정된다.
- 커널의 크기를 작게 설정했을 때는 고주파 성분 손실이 적고 처리 속도가 빠르다. 잡음 제거가 효과적이지 않다.
- 커널의 크기를 크게 설정했을 때는 잡음 제거에 효과적이고 더 부드러운 블러링 효과를 보인다. 고주파 성분을 많이 제거하여 세부 정보 손실 초래, 처리 속도가 느리다.
- 작은 커널은 엣지 검출에 활용되며 큰 커널은 노이즈 제거와 블러링에 활용 된다.

문제5. BMP 이미지 파일은 signed char과 unsigned char중에 어떤 걸 사용하는지 서술하고, 왜 그걸 쓰는지 이유도 서술하시오.

- unsigned char.

- 영상 처리는 음수가 없기 때문. RGB 값을 표현. RGB는 0~255까지 256개의 값을 다룰수 있음. 이는 8비트(1바이트)로 표현.

문제6. BMP는 (A)와 (B)로 나뉜다. 이때 A와 B에 들어갈 명칭을 작성하고, A와 B를 설명하시오.

- A : DDB(장치 의존 비트맵) - 모니터(출력 장치)에 나타나는 영상이 모니터의 화면에 설정된 값에 의해 출력되는 방식
- B : DIB(장치 독립 비트맵) - 출력 장치의 설정이 달라지더라도 자신의 비트맵 값이 그대로 출력되도록 하는 방식

문제7.

> openCV에서 RGB 이미지를 로드하여 처리할 때 기본적으로 색상 순서를 ⓐ____ 순으로 처리한다.
> 이는 영상 처리 과정 중 코드의 복잡성 야기하기 때문에 미리 ⓑ___ 형식으로 바꾸어 가독성을 높이는 과정을 거치는게 좋다.
> ⓐ와 ⓑ에 들어갈 내용을 채워넣은 후,
> 아래의 ①~⑤ 중 이 동작을 수행하는 소스코드의 나머지 부분을 고르고 소스코드의 빈칸 ⓒ____ 을 채우시오.
>
> 「ⓐ _____ⓑ _____ⓒ _____
>
> ①~⑤ 중 옳은 소스 코드 번호_____」

① cv2.cvtColor(image, cv2.TrueCOLOR_『 ⓒ 』)

② cv2.cvtColor(image, conv.COLOR_『 ⓒ 』)

③ cv2.cvtColor(image, ch2.COLOR_『 ⓒ 』)

④ cv2.cvtColor(image, cv2.COLOR_『 ⓒ 』)

⑤ cv2.cvtColor(image, cv2.PIGMENT_『 ⓒ 』)

- a => b, g, r
- b => r, g, b
- c => 4번

문제8. 평균 필터링의 정의에 대해 서술하고 3 X 3 평균 커널과 5 X 5 평균 커널을 그려 크기에 따른 스무딩의 특성을 설명하시오

- 입력 영상의 픽셀 밝기값을 주변 픽셀들의 밝기값의 평균으로 대체하여 영상을 스무딩(smoothing)시키면서 잡음을 줄이는 필터링 기법
- 잡음과 경계 부분을 제거하여 부드러운 영상으로 변환하는 특성을 가지고 있음

문제9. 명암 대비에 따른 영상의 특징을 적고, 변환 함수의 기울기가 1보다 큰 경우와 1보다 작은 경우 히스토그램이 어떻게 변하는지 서술하시오.

- 낮은 명암 대비를 가진 영상, 높은 명암 대비를 가진 영상, 좋은 명암 대비를 가진 영상

- 변환 함수의 기울기가 1보다 큰 경우는 히스토그램이 확장된다
- 변환 함수의 기울기가 1보다 작은 경우는 히스토그램이 축소된다

문제10. HSV 색상 모델과 RGB 색상 모델의 차이점을 서술하고 HSV 색상 모델의 의미를 서술하시오.

- RGB는 레드, 그린, 블루로 빛의 3원색인 가산조합으로 나타낸다. HSV는 색상, 채도, 명도로 나타낸다.
- RGB 색상 모델이 인간의 색상 정보 체계와는 다르다는 문제점을 해결하고, 인간이 인식하는 색상과 흡사한 색상 모델을 만들기 위해 고안된 색상 모델이다

문제11. 적응적 임계값을 적용할때 기본 전제 과정을 서술하고 코드 dh_rest = np.int(gray_img.shape[0]%N) 의 의미를 설명하시오.

- 작은블록은 대략적으로 균등한 조명을 가질 가능성이 크기 때문에 임계값 적용에 보다 적합하다.
- 블록의 크기보다 작은 영상의 가장자리 영역을 연산에서 제외시킨다.

문제12. 영상에서 전역 임계값을 이용하여 영상 분리 및 배경 분할 등을 진행할 때의 한계점(분리가 잘 이뤄지지 않는 경우)을 히스토그램상의 특징과 관련하여 서술하시오. 그와 같은 경우 전역 임계값 적용대신 사용할 수 있는 방법에 대해 쓰고, 해당 방법을 간단히 설명하시오.

- 영상 안에 조명이 일정하지 않거나 다양한 색상을 가지는 전경 물체가 있을 경우 한계를 가진다.
- 히스토그램에서 객체와 배경 간의 픽셀 값이 명확하게 구분되지 않기 때문이다.
- 적응적 임계 적용 방법을 사용한다.
- 적응적 임계 적용 방법은 영상을 일정 블록으로 분할하고 각 분할된 블록마다 제각기 다른 임계값을 적용한다

문제13. cv2.calcHist(images, channels, mask, histSize, ranges[, hist[, accumulate]])에서 사용되는 파라미터들에 대한 설명을 쓰시오

- images - uint8, float32 유형의 소스 이미지를 나타낸다. 대괄호 묶어 사용
- channels - 히스토그램을 계산하는 채널의 인덱스. 대괄호로 사용
- mask - 마스크 영상
- histSize - 히스토그램 bin 개수를 나타내며 대괄호로 묶어 사용
- ranges - 색상 범위를 나타낸다

문제14. 히스토그램의 의미와 히스토그램 스트레칭의 방법에대해 설명하시오

- 영상 안에 포함된 각각 다른 밝기값을 갖는 픽셀들의 수(분포)를 보여주는 그래프를 의미
- 영상의 밝기값 범위를 확장(또는 축소) 시킴으로써 영상의 대비(constrast)를 향상시키는 방법

문제15. 가우시안 노이즈가 포함된 이미지가 있다고 가정하고, 이 이미지에 대해 평균값 필터링과 중간값 필터링을 각각 적용했을 때, 질문에 답하시오

- 두 필터링 방법을 적용한 후, 노이즈 제거 성능의 차이를 설명하시오
    - 평균값 필터링은 영상의 선명도가 유지되며 적절하게 잡음이 제거되었음을 알 수 있다
    - 중간값 필터링은 잡음의 정도가 심한 영상의 경우 평균 필터링처럼 잡음을 효과적으로 감소시키지 못하고 있음을 알 수 있다
- 두 필터를 이미지의 경계(에지) 보존 측면에서 비교하시오

    - 중간값 필터는 에지와 경계선과 같은 영상의 고주파 성분을 잘 보존하면서 잡음 제거할 수 있다는 장점이 있다

문제16. 소금&후추(Salt&Pepper) 잡음이 있는 영상이 있다 이걸 평균 필터링(Average Filtering)으로 하면 안 되는 이유를 서술하시오

- 잡음 픽셀의 밝기값이 주변 픽셀의 밝기값 결정을 위한 평균 계산에 큰 영향을 주기 때문에 전체적으로 잡음이 제대로 제거되지 않기 때문이다

문제17. 아날로그-디지털 변환기(ADC)는 포토 다이오드에 입력된 아날로그 전기신호를 0/1의 디지털 신호로 바꿔야 한다. 아날로그-디지털 변환기에서 샘플링(Sampling)과 양자화(Quantization)라는 중요한 두 가지 작업이 필요하다. 샘플링과 양자화에 대해 자세히 설명하라.

- 샘플링 - 연속적인 아날로그 신호로부터 일정한 간격의 주기로 신호의 강도를 수집하는 것을 말한다
- 양자화 - 각 샘플링 단계에서 얻어진 신호값을 일정한 범위의 디지털 값으로 매핑시키는 과정이다

문제18. 히스토그램은 이미지의 밝기 값 분포를 나타내는 그래프이다. 이미지의 대비를 향상시키기 위해 히스토그램을 평탄하게 만드는 기법은 무엇이며 이 기법에 대한 효과와 OpenCV에서 불러오기 위한 함수를 적으시오.

- 히스토그램 평활화. 히스토그램 분포를 정규 분포 형태의 히스토그램 갖도록 하여 화질을 개선시킨다.
- equalizeHist 함수

문제19. 빈칸 (1)에 들어갈 숫자와 입력한 수치가 방지하고자 하는 문제와 사용한 방법에 대해 설명하시오.

```
for h in range(RGB_img1.shape[0]):
    for w in range(RGB_img.shape[1]):
       r = np.int32(R_img1[h,w]) + np.int32(R_img1[hw])
       if(r> (1) ):
            R_plus[h,w] = (1)
       else:
            R_plus[h,w] = r
```

- (1) - 255
- 오버플로우 발생 방지
- Saturation - 픽셀값이 설정된 최대값을 초과할 경우 초과된 값을 미리 설정된 최대값(255)으로 교체한다

문제20. 아래 코드에 대해 설명하고 픽셀 빼기 연산에 대한 설명과 사용하는 이유를 설명하시오.

```
R_plus[h,w] = np.abs(np.int32(R_img1[h,w]) - np.int32(R_img2[h,w]))
```

- 뺄셈의 경우 결과값이 음수가 될 수 있으므로 절대값을 취해준다.
- 두 영상 간에 변화를 감지할 수 있다. 그래서 변화된 부분을 검출해 낼 수 있다.

문제21. RGB 컬러 모델과 YCbCr 컬러 모델을 비교하여 설명하시오

-

문제22. 이진 영상의 경우인 픽셀 반전연산과 컬러 영상의 경우 픽셀 반전연산을 서술하시오. 영상 A가 있고, 이 영상의 반전 영상인 B를 얻어야 하는 경우, 얻는 방법과 수식을 서술하시오.

- 이진 영상의 경우 -> 검은색 픽셀은 흰색으로, 흰색 픽셀은 검은색으로 반전시킨다.
- 컬러 영상의 경우 -> 각 픽셀 값의 보수로 매핑시킨다
- 입력된 한 영상 P의 각 좌표값을 255에서 빼고 그 결과 값을 O(x, y)에 입력함으로써 반전 영상을 얻을 수 있음
- O(x, y) = 255 - P(x, y)

문제23. 픽셀 처리 연산자 중에서 영상의 밝기(명암도)를 높여 주기 위한 연산 두 가지는 무엇이며, 또 이 두 연산자의 차이점은 무엇인가 ?

- 픽셀 더하기, 픽셀 곱하기
- 덧셈 연산은 영상의 전반적인 밝기만 높인다. 명암 대비 조절 못함
- 곱셈 연산은 밝기 뿐만 아니라 명암 대비도 조절한다.

문제24. 3x3 평균 커널을 사용하여 블러 처리를 할 때 5x5 평균 커널의 블러 처리와 비슷한 결과를 낼 수 있는 방법이 있다. 이 방법을 서술하시오.

- 3x3 커널을 여러 번 블러 처리를 하면 비슷한 결과를 낼 수 있다.

문제25. Lowpass, Highpass, Bandpass에 대해 설명하고 영상에 평균 값 필터를 3x3 크기의 마스크로 적용 했을 때 Lowpass, Highpass의 커널을 그리시오

- Lowpass - 저주파수 성분을 통과시키고, 고주파수 성분을 차단
- Highpass - 주파수 성분을 통과시키고, 저주파수 성분을 차단
- Bandpass - 특정 주파수 대역만을 통과시키는 필터

- Lowpass $\frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$
- Highpass $\begin{bmatrix}-1 & -1 & -1\\ -1 & 8 & -1\\ -1 & -1 & -1\end{bmatrix}$

문제26. 다음은 이미지에 워터마크를 적용하기 위한 코드의 일부입니다. 워터마크가 제대로 작동하지 않거나 전혀 보이지 않는 상황입니다. 이 워터마크의 문제와 이유에 대해 서술하시오

```
gWaterMarker = gImg1.copy()
for h in range(gImg1.shape[0]):
    for w in range(gImg1.shape[1]):
```

```
    imgVal = gImg2[h, w]
        if(imgVal == 128):
             # 최하위 비트가 0
            if(gImg1[h, w] % 2 == 0):
                gWaterMarker[h, w] = gWaterMarker[h, w] + 1
            else:
            # 최하위 비트에 워터마크 세팅
                gWaterMarker[h, w] = gWaterMarker[h, w] - 1
cv2_imshow(gWaterMarker)
```

상황
1.원본 이미지(gImg1)와 비교 이미지(gimg2)가 있습니다. 워터마크는 gimg2의 특정 픽셀 값(예: 128)에 따라 gImg1에 적용됩니다.

2.워터마크 적용 조건은 gimg2에서 픽셀 값이 128인 경우, gImg1의 해당 픽셀이 짝수일 때 1을 추가하고, 그렇지 않으면 워터마크를 제거합니다.

워터마크가 제대로 적용되지 않거나 전혀 보이지 않는 상황이 발생하고 있습니다.

- 답 -> 적용범위와 가시성의 제한의 문제가 있다
- gimg2에서 특정값(128)만 대상으로 하여 전용 픽셀이 적고, 최하위 비트만 변경되어 시각적으로 인식하기 어렵다

문제27. 영상 처리에서 히스토그램 스트레칭과 히스토그램 평활화는 화질을 개선하는 두 가지 중요한 기법입니다. 히스토그램 스트레칭과 히스토그램 평활화에 대해 각각 설명하고, 두 기법이 영상의 대비를 개선하는 방식의 차이를 서술하시오.

- 히스토그램 스트레칭 -> 영상의 밝긱값 범위를 확장(또는 축소) 시킴으로써 영상의 대비를 향상시키는 방법
- 히스토그램 평활화 -> 비선형적인 함수를 이용하여 영상으로 하여금 정규 분포 형태의 히스토그램을 갖도록 하여 화질을 개선하는 방법
- 두 기법의 차이 -> 스트레칭은 선형적인 함수를 이용하고, 평활화는 비선형적인 함수를 이용한다.

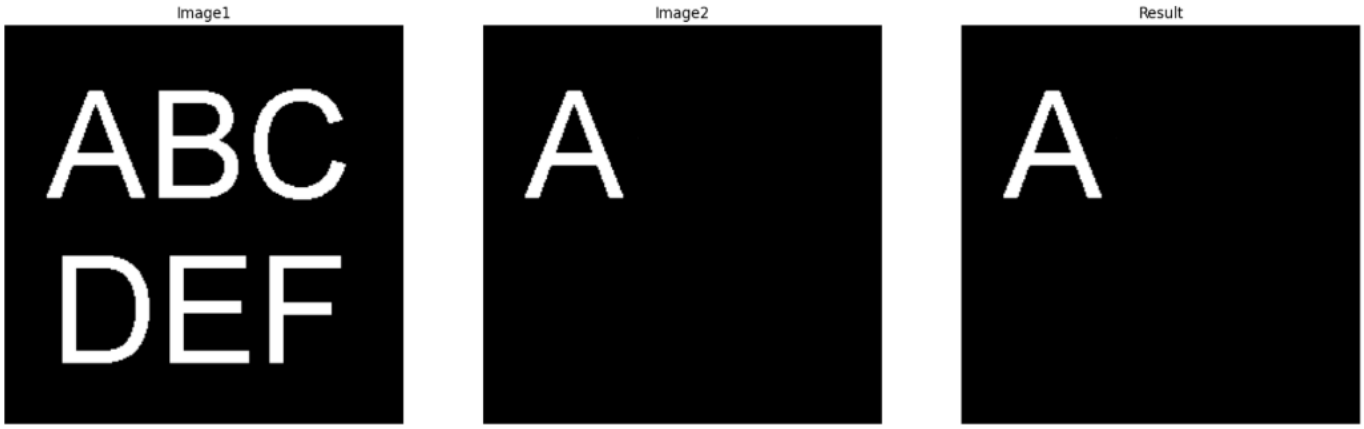
문제28. 디지털 영상처리에서 많이 사용되는 영상 파일포맷들 종류와 그것들에 대해 상세히 설명하시오.

-

문제29. 두 개의 이진 영상을 입력 받아서 비트 연산을 수행한 결과를 출력한 것이다. 아래의 코드에서 빈칸에 들어가야 하는 것은 무엇이고 코드의 역할은 무엇인지 쓰시오.

```
def saturation(value):
  if(value>255):
    value = 255;
  return value

for h in range(RGB_img1.shape[0]):
  for w in range(RGB_img1.shape[1]):
    R_plus[h,w] = saturation(
    G_plus[h,w] = saturation(
    B_plus[h,w] = saturation(
```




```
np.int32(R_img1[h, w] & np.int32(R_img2[h, w]))
np.int32(G_img1[h, w] & np.int32(G_img2[h, w]))
np.int32(B_img1[h, w] & np.int32(B_img2[h, w]))
```

- AND 연산으로 두 영상 모두 같은 픽셀로 표현된 픽셀은 결과 영상에서 같은 값으로 표시한다. 그렇지 않으면 사라진 것과 같은 효과를 나타낸다.

문제30.

다음과 같은 영상 파일 (ㄱ), (ㄴ)이 있다.

(1) 픽셀 더하기 연산을 응용하여, (ㄱ)은 gImg1, (ㄴ)은 gImg2라는 변수명으로 불러왔을 때 (ㄷ)과 같은 결과물을 출력하려 한다.
이에 대한 반복문을 작성하시오(단, 출력 배열 gPlus에 결과를 저장하여 출력한다. 출력 배열 gPlus는 gImg와 동일한 크기의 unsigned byte 타입 배열로, 미리 생성되어 0으로 초기화되어 있다고 가정한다.).

(2) 픽셀 더하기 연산을 응용하여 페이드인/페이드아웃 기능을 구현하는 방법에 대해 설명하고 간단한 예시를 보이시오((1)과 같은 영상 파일 gImg1, gImg2, 같은 출력 배열 gPlus를 이용한다.).

- gPlus = np.zeros((gImg.shape[0], gImg.shape[1]), dtype=np.ubyte) // 이미 되어 있음

```
## (1)
for h in range(gImg1.shape[0]):
    for w in range(gImg1.shape[1]):
        plusVal = np.int32(gImg1[h, w]) + np.int32(gImg2[h, w])
        if plusVal > 255:
            plusVal = 255
        gPlus[h, w] = plusVal
```

```
## (2)
weight = 0.7

for h in range(gImg1.shape[0]):
    for w in range(gImg1.shape[1]):
        plusVal = np.int32(gImg1[h, w])*weight + np.int32(gImg2[h, w])*(1-weight)
        if(plusVal > 255):
            plusVal = 255
        gPlus[h, w] = plusVal
```

- 픽셀 결합 방법을 이용한다. 이 방법은 두 개의 입력 영상이 가중치에 의해 결합되어 하나의 결과 영상을 만든다.
- 가중치인 weight를 0에서 1로 점진적으로 증가시키면 페이드인 기능이 되고, weight를 1에서 0으로 점진적으로 증가시키면 페이드아웃 기능이 된다.
- 0과 가까우면 첫번째 이미지가 더 반영되고 1과 가까우면 두번째 이미지가 더 반영된다.