2020 September 20

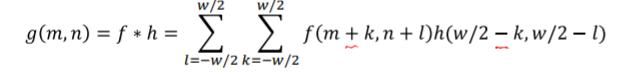
**20181593 계인혜**

**고급 소프트웨어 실습 2주차 과제**

Average filtering & Median filtering’s efficient implementation method

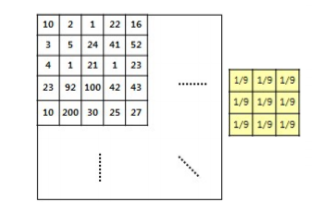
1. Average 필터링을 효율적으로 구현할 수 있는 Moving Average에 대하여 공식을 이용하여 설명한다.

Average 필터는 픽셀 값에 대하여 주변의 픽셀 값들과의 평균 값을 구하여 결과 이미지에 할당하는 방식으로 작은 노이즈를 제거하는 효과가 있다. Average 필터링 공식은 다음과 같다.

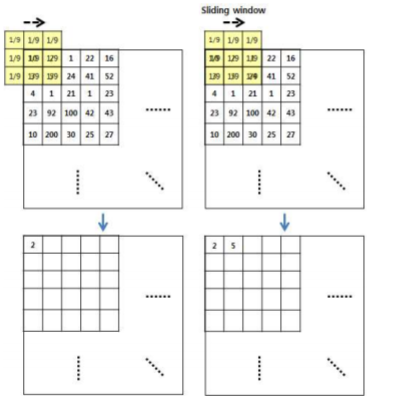


Average 필터의 적용 방식은 다음 그림과 같다. 아래와 같은 (a) 33 uniform 커널을 오른쪽 그림에 적용한다고 하자. 계산과정은 아래의 그림에서 확인할 수 있다.

시계이(가) 표시된 사진

자동 생성된 설명

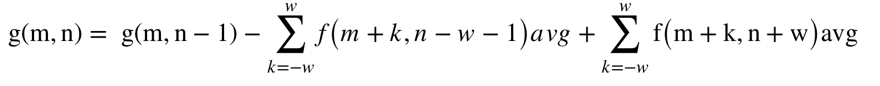
f h



**g**

그런데 이 과정에서 중복된 연산을 상당히 많이 수행하는 것을 확인할 수 있다. 예를 들어, g(1,1) (g의 1행 1열)을 계산하기 위해서 f(1,1), f(1,2), f(2,1), f(2,2)이 필요하다. 그 다음 g(1,2)를 계산하기 위해서 f(1,1), f(1,2), f(1,3), f(2,1), f(2,2), f(2,3)이 필요하다. 이렇게 중복으로 계산되는 문제를 해결하기 위한 방법으로 Moving Average 연산법이 있다.

Moving Average 연산법에 대하여 알아보자. Moving Average 연산은 기존에 계산했던 값을 기준으로 새롭게 추가되는 부분은 더하고, 기존 값에서 불필요한 부분은 빼주면 된다. 위의 예시를 Moving Average로 구현한다면, Average와 마찬가지로 g(1,1)을 계산하기 위해서 f(1,1), f(1,2), f(2,1), f(2,2)이 필요하다. 그러나 g(1,2)를 계산할 때, 앞에서와 달리 g(1,1)의 결과에서 새롭게 필요한 f(1,3), f(2,3)을 추가하기만 하면 되는 것이다. 이를 공식화하면 다음과 같다.



1. Median 필터링을 효율적으로 구현할 수 있는 방법에 관하여 도식적으로 설명해 보자. 3x3 윈도우 사이즈를 사용하는 경우를 가정하여 구체적으로 설명한다.

Median 필터링은 Average 필터링과 비슷하게 필터 윈도우를 이동시켜 가며, 윈도우 안의 픽셀 값들의 중간 값을 결과 이미지에 할당한다. 이 방식은 Salt & Pepper 타입의 작은 노이즈들을 제거하는데 탁월하며, 이미지의 sharpness도 보존할 수 있다. 또한, Median 필터링은 커널 또는 Convolution 연산으로 표현할 수 없고 linear 연산도 아니다. Median 값을 구하기 위해서는 필연적으로 Sorting이 필요하다. 이로 인해 Average 필터링과 같은 Convolution 기반의 필터링보다 처리 속도가 느리다는 특징이 있다.

다음 예시를 보며 Median 필터링의 효율적 구현방법을 생각해보자.

그리기, 물, 옅은, 방이(가) 표시된 사진

자동 생성된 설명 물, 옅은, 커플, 구름낀이(가) 표시된 사진

자동 생성된 설명

**그림2**

**그림1**

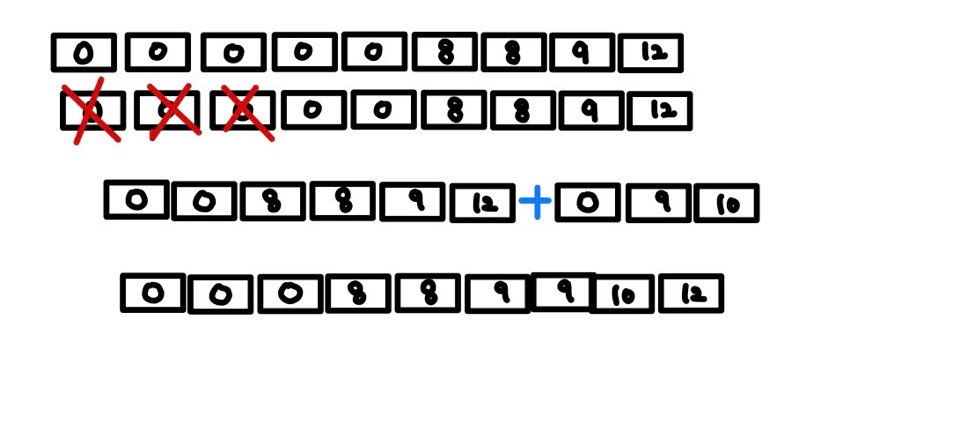
그림1과 같은 이미지가 있다. 이를 f라고 하도록 하자. 그림2는 f(1,1)의 Median 필터링을 수행하는 과정이다. 패딩을 0으로 설정하여 정렬(ascending)하면 0, 0, 0, 0, 0, 8, 8, 9, 12로 Median 값은 0이다. 이어서 그림 3과 같이 f(1,2)를 정렬하면 0, 0, 0, 8, 8, 9, 9, 10, 물, 시계, 옅은, 앉아있는이(가) 표시된 사진

자동 생성된 설명12로 Median 값은 8이다. Average 필터링과 마찬가지로 중복된 픽셀 값을 여러 번 정렬하는 것을 확인할 수 있다. 이를 방지하기 위해서는 이전의 결과 값을 기준으로 불필요한 픽셀 값은 제외하고, 새로 들어오는 픽셀들을 추가하여 정렬하면 된다.

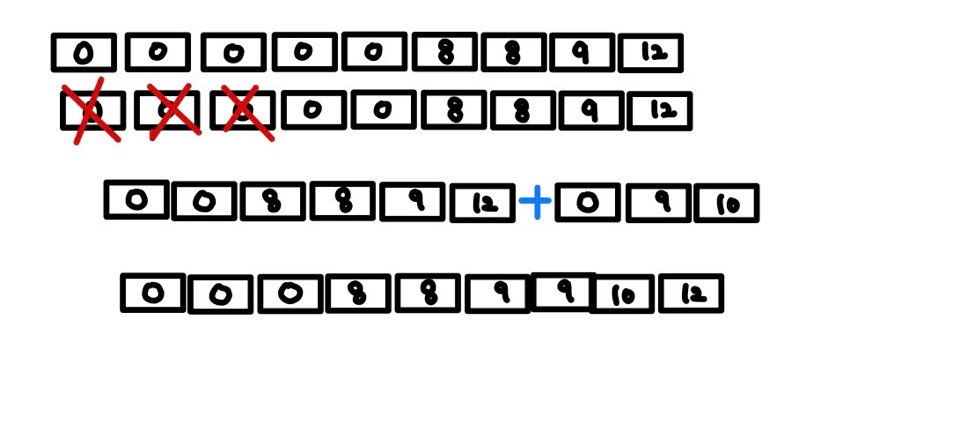
**그림3**

직접 그린 그림을 보며 조금 더 구체적으로 살펴보자.

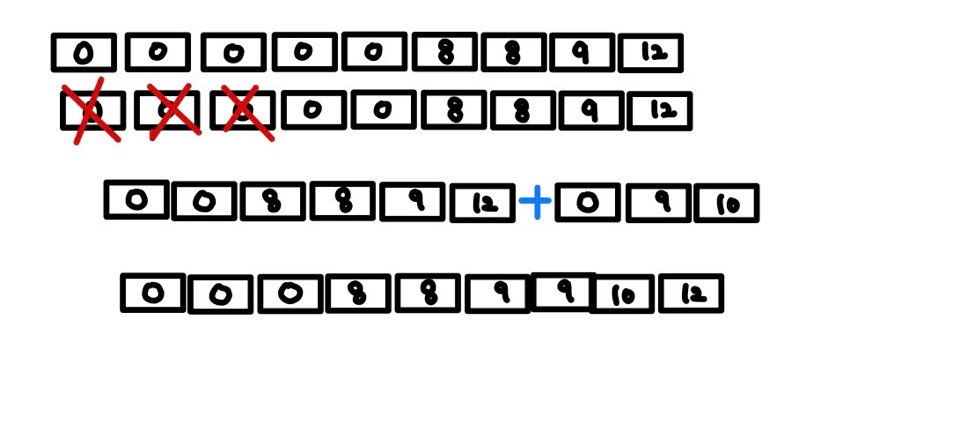
① 처음 linked-list에 그림2의 필터 값들이 오름차순으로 정렬되어 있다.



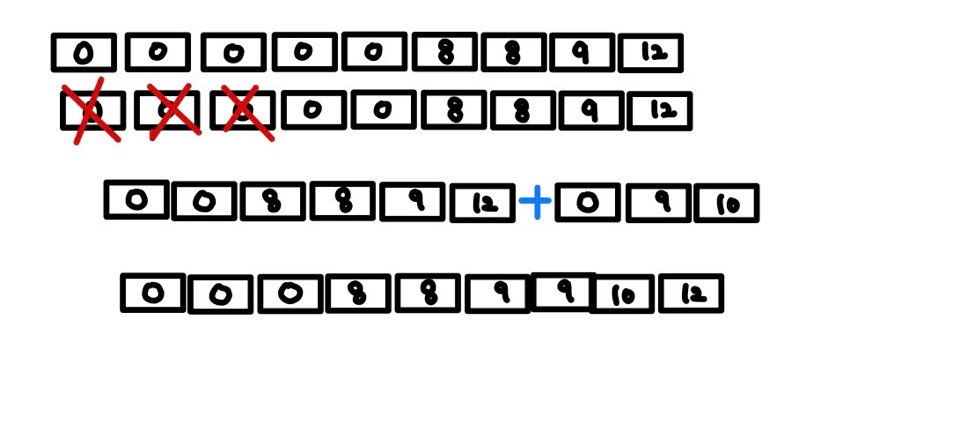
② 필터가 오른쪽으로 한 칸 움직이며 불필요한 부분을 삭제한다. (0,0,0)



③ 필터가 오른쪽으로 한 칸 움직이며 새로운 부분을 삽입한다. (0,9,10)



④ 새로 삽입된 부분도 오름차순에 맞춰 정렬한다.



③의 그림에서 기존의 6개는 이미 정렬되어 있는 상태이기 때문에 새로 삽입되는 세개의 원소만 올바른 자리에 넣어주면 된다. 다음과 같은 방법으로 Median 필터의 효율적인 구현을 할 수 있다.