

영상처리

Noise Reduction

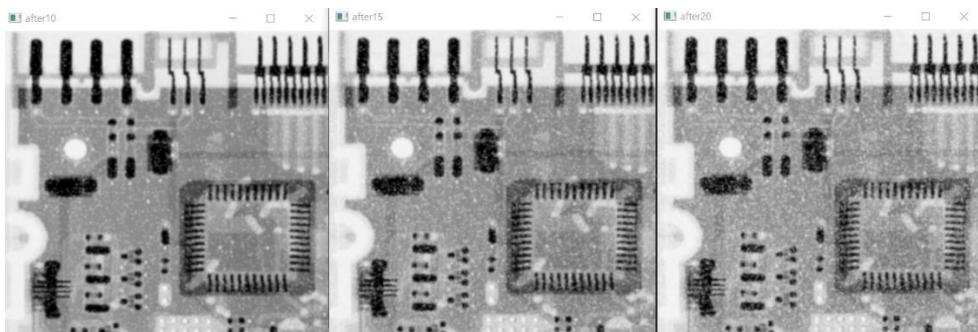
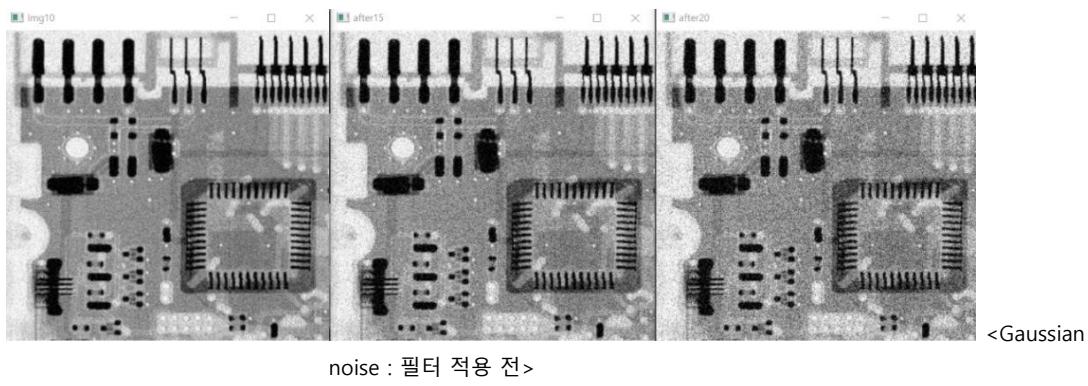
1. Gaussian Noise

1-a. Gaussian Filter

Gaussian Noise 제거를 위해서 노이즈의 정도가 다른 3 가지 이미지에 대해서 여러가지 필터를 적용하면서 관찰했다. 우선 Gaussian Filter 를 적용했다.

	0.10	0.15	0.20
PSNR	71.5131	68.1202	65.7333

<필터링 하기 전 PSNR 수치>



왼쪽부터 차례로 노이즈 정도가 1.0, 1.5, 2.0 인 이미지에 필터를 적용한 결과의 이미지이다. 육안으로는 큰 차이가 느껴지지 않았지만, PSNR 수치를 비교했을 때, 살짝 개선된 수치를 관찰할 수 있었다. 시그마의 크기를 1,2,5 로 변경하면서 노이즈를 관찰했다. 시그마의 크기가 1 일 때, 가장 개선된 수치를 얻을 수 있었고, 시그마의 크기가 커질수록 PSNR 의 수치가 작아지는 것을 확인 할 수 있었다. 아래는 전체적인 결과에 대한 표이다.

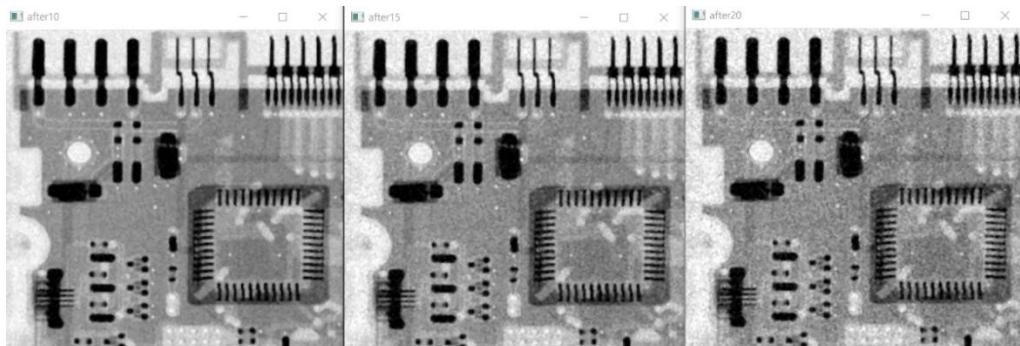
Gaussian	0.10(PSNR)	0.15(PSNR)	0.20(PSNR)	소요시간(ms)
시그마 : 1	73.2656	70.4332	68.2389	0
시그마 : 2	69.8128	68.5325	67.1435	0
시그마 : 5	65.5786	65.0739	64.4254	2

<Gaussian Filter>

필터링 하기 전 PSNR 수치를 비교했을 때, Gaussian filter 의 경우에는 Gaussian noise 제거에 시그마의 크기가 1 일 때, 가장 효과적이라는 사실을 확인할 수 있다.

1-b. Median Filter

Gaussian Noise 에 Median Filter 를 적용했을 때를 확인해 보았다.



<Median Filter : window size(3x3)>

Median	0.10(PSNR)	0.15(PSNR)	0.20(PSNR)	소요시간(ms)
3x3	76.1473	73.7879	71.9055	0
5x5	73.8469	72.9189	72.0049	1

<Median Filter>

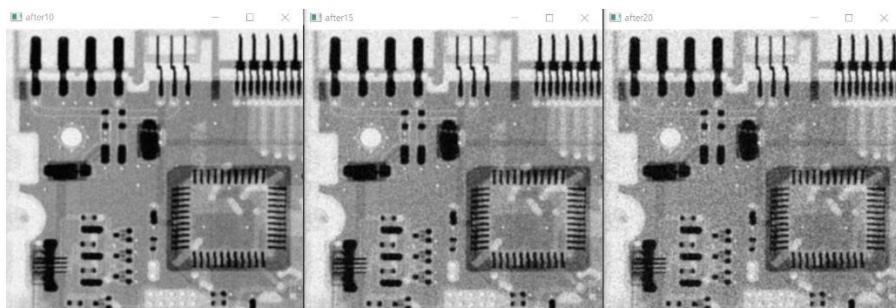
Median filter 역시 윈도우 크기가 커질수록 PSNR 의 값이 줄어드는 경향성을 확인할 수 있었다.

1-c. Arithmetic Filter

아래는 Arithmetic Mean Filter 를 적용했을 때의 표와 이미지이다.

Arithmetic	0.10(PSNR)	0.15(PSNR)	0.20(PSNR)	소요시간(ms)
3x3	75.8401	74.2483	72.7066	1
5x5	71.927	71.5102	70.9911	4
7x7	69.2763	69.0936	68.8545	9

<Arithmetic Filter>



<Arithmetic Mean Filter : window size(3x3)>

Arithmetic Mean Filter 역시 윈도우 사이즈 3 일 때, 가장 높은 PSNR 추치를 관찰할 수 있었다. 원

도우 사이즈가 커질수록 처리시간의 소요가 길어지고 PSNR 수치 역시 줄어드는 것을 관찰할 수 있었다.

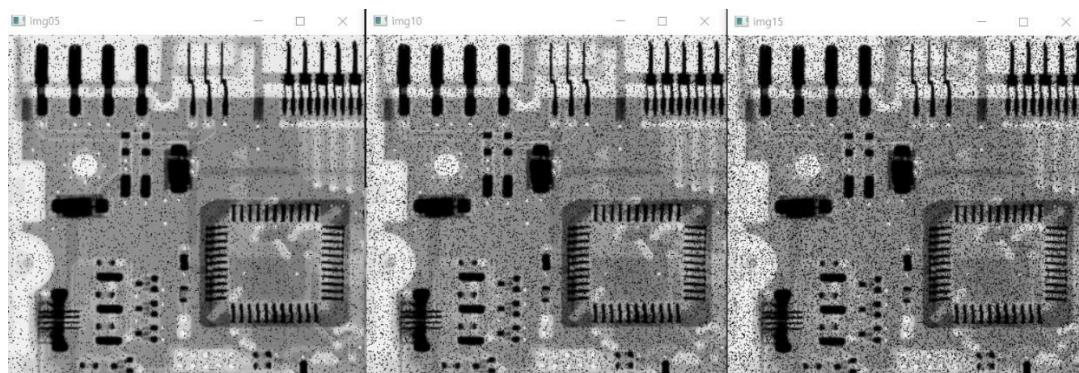
1-d. Gaussian Noise 제거 결론

Gaussian Noise 를 제거하기 위해서 Gaussian Filter, Median Filter, Arithmetic Filter 를 윈도우 사이즈를 달리해서 적용해 봤다. PSNR 수치상으로는 개선되는 것을 확인할 수 있었지만, Gaussian Filter 의 경우에는 시각적으로는 노이즈 제거에 큰 변화를 확인할 수 없었다. Median Filter 와 Arithmetic Filter 는 유사한 PSNR 수치를 보여줬지만, 연산 속도를 고려해 본다면, 윈도우 크기 3 인 Median Filter 가 Gaussian Noise 제거에 가장 효과적이라고 생각한다.

2. Pepper Noise

2-a. Max Filter

Pepper Noise 는 이미지에 무작위로 가장 어두운 값의 픽셀이 나타나는 노이즈이다. Max Filter 를 적용하면 주변의 값 중에서 가장 큰 값을 해당 픽셀의 값으로 지정하기 때문에 Pepper Noise 를 효과적으로 제거할 수 있다.

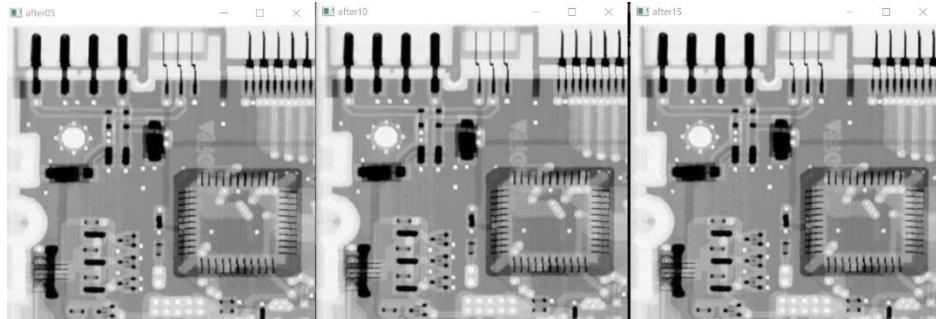


<Pepper Noise : Filtering 하기 전>

Pepper	0.05	0.10	0.15
PSNR	65.2575	62.2229	60.4360

<필터링 전 PSNR>

Pepper Noise 에 대해서 Max Filter 를 윈도우 사이즈를 변경해서 관찰했다.



<Max Filter : window size(3x3)>

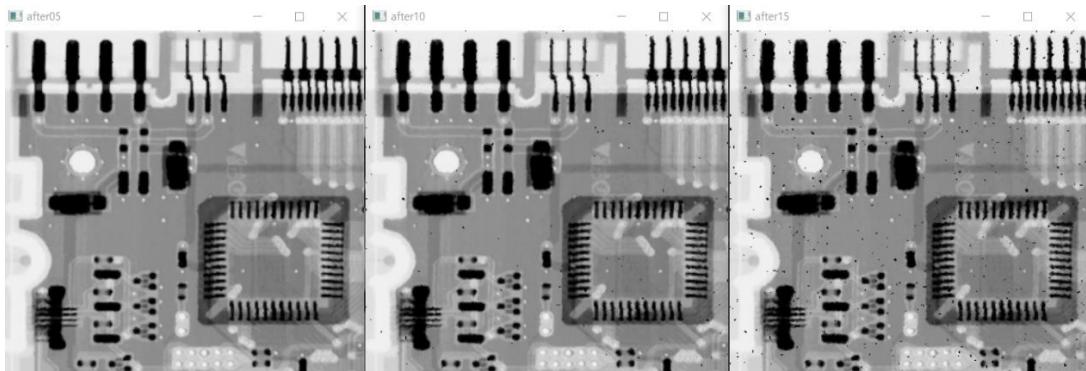
Pepper Noise에 Max Filter의 윈도우 사이즈를 변경하면서 적용해본 결과는 아래와 같다.

Max	0.10(PSNR)	0.15(PSNR)	0.20(PSNR)	소요시간(ms)
3x3	67.6385	67.7159	67.7874	2
5x5	63.2625	63.2946	63.3277	5
7x7	61.4123	61.4341	61.4544	7

다른 Filter와 동일하게 윈도우 사이즈가 3일 때, 가장 나은 PSNR 값을 얻을 수 있었다. Max Filter의 특성상 이미지의 전반적인 밝기가 밝아진 것을 확인할 수 있다.

2-b. Median Filter

아래는 Median Filter 적용한 이미지(window size : 3x3)와 그에 따른 데이터이다.



<Median Filter : window size(3x3)>

왼쪽부터 차례로 0.05, 0.10, 0.15의 노이즈의 정도를 갖는 이미지에 대한 필터링 결과이다.

Median	0.10(PSNR)	0.15(PSNR)	0.20(PSNR)	소요시간(ms)
3x3	79.1706	75.6151	71.8359	0
5x5	74.7492	73.6925	72.3774	1

<Median Filter>

Max Filter와 비교했을 때, 상당히 개선된 PSNR 수치를 얻을 수 있었다. Max 필터와 비교했을 때, 연산속도 면에서 큰 차이를 관찰할 수 있었다. 이론적으로 Max Filter는 윈도우 사이즈에 해당하는

픽셀 중에서 가장 큰 픽셀 값을 찾으면 되기 때문에 정렬이 필요한 Median Filter 보다 연산량이 적어서 더 빨라야 하지만, OpenCV에 구현되어 있는 Median Filter의 최적화가 잘 되어 있기 때문에 Max Filter가 더 오래 걸린 것 아닐까 추측한다.

2-c. Pepper Noise 제거 결론

Pepper Noise 제거에는 수치적, 시간적으로 윈도우 사이즈가 3인 Median Filter가 Pepper Noise 제거에 가장 효과적이었다. 개인적으로는 Max Filter가 Median Filter 보다 시작적으로 보기에는 좋았지만, 픽셀의 밝기가 전반적으로 밝아지기 때문에 높은 PSNR 수치를 얻을 수 없었다. 그러나, Median Filter는 전반적인 이미지의 밝기에 큰 차이가 없으면서, 주변 픽셀 중 중앙값을 취함으로써 outlier를 효과적으로 제거할 수 있었다.

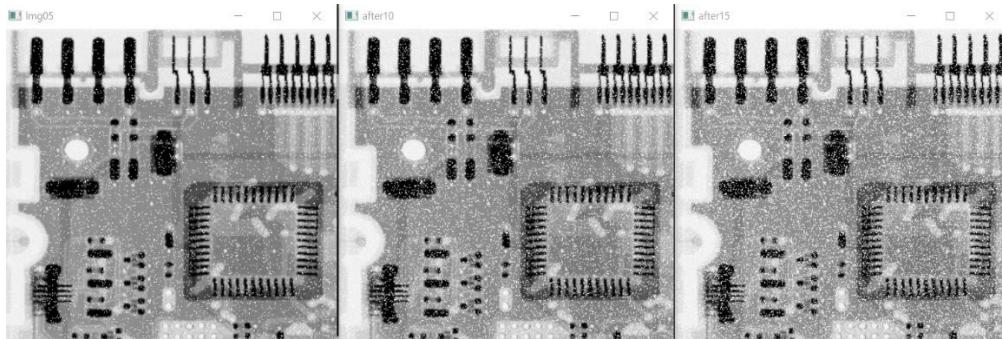
3. Salt Noise

3-a. Min Filter

Salt Noise를 제거하기 위해서 윈도우 크기를 달리해서 Min Filter를 적용했다. 필터링 하기 전 PSNR 수치와 이미지는 아래와 같다.

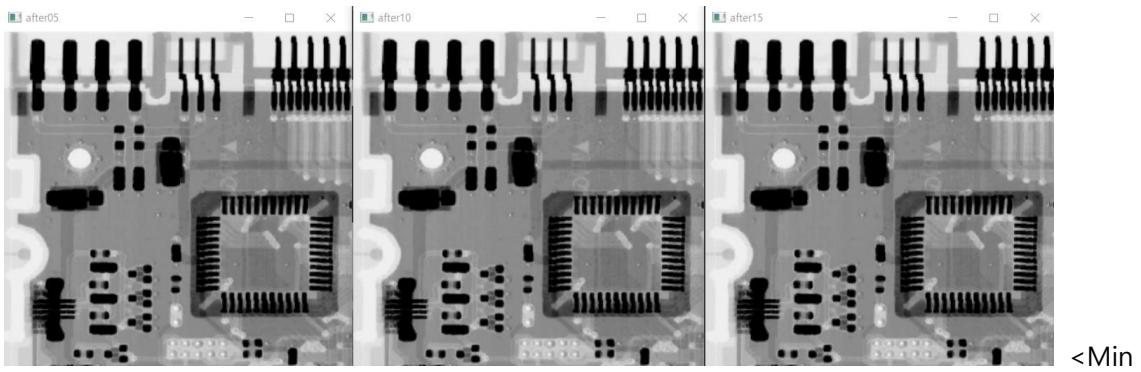
Salt	0.05	0.10	0.15
PSNR	67.3986	64.3479	62.5754

<Salt Noise : PSNR>



<Salt Noise : 필터 적용 전>

Salt Noise의 제거에 효과적이라고 알려져 있는 Min Filter를 적용한 결과는 아래와 같다.



Filter : window size(3x3)>

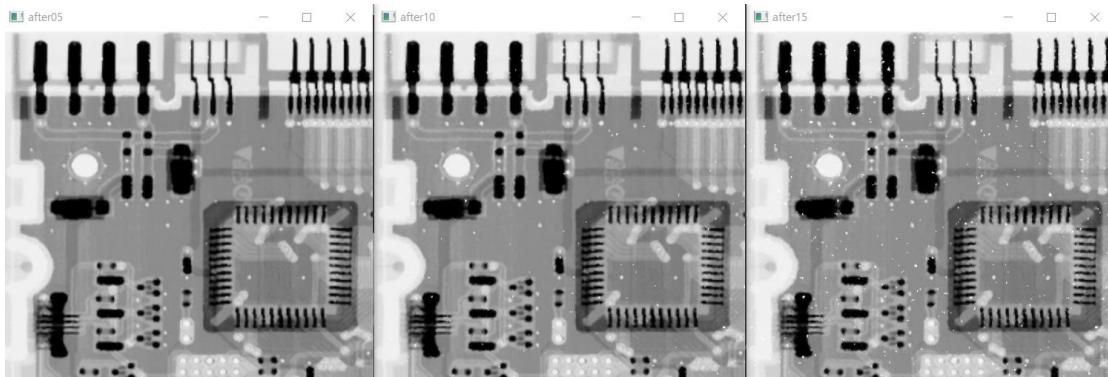
Min Filter 를 적용한 전과 후의 이미지를 시각적으로 관찰했을 겨우, 대부분의 Salt Noise 가 제거된 것을 관찰할 수 있다. 처리 후의 PSNR 수치 역시 개선이 되었다. 노이즈의 정도가 0.05 일 때, 수치에 큰 변화가 없었지만, 0.10 와 0.15 일 때는 유의미한 변화를 관찰할 수 있었다. 대신, 윈도우 사 이즈가 3 보다 커지는 경우에는 오히려 PSNR 수치가 필터링을 하기 전보다 감소했다.

Min	0.05(PSNR)	0.10(PSNR)	0.15(PSNR)	소요시간(ms)
3x3	67.4543	67.5210	67.6035	2
5x5	62.6481	62.6895	62.7290	4

<Min filter : PSNR>

3-b. Median Filter

Median Filter 는 window 전체에서 중앙값을 찾아, 중앙값을 해당 픽셀의 값으로 사용하는 Filter 이다. 아래는 Median filter 를 진행했을 때의 이미지와 PSNR 수치에 대한 표이다.



<Median Filter : window Size(3x3)>

Median	0.10(PSNR)	0.15(PSNR)	0.20(PSNR)	소요시간(ms)
3x3	79.2866	76.2408	73.0115	0
5x5	74.2821	72.9343	71.4606	1

<Median Filter : PSNR>

시각적으로는 Median Filter 보다 Min Filter 가 Salt Noise 제거에 효과적으로 보인다. 하지만, PSNR이라는 수치를 비교했을 때는 Median Filter 가 Min Filter 보다 더 높은 값을 얻을 수 있었다. 성능 면에서도 월등한 수치를 보였다. 이론적으로 Median Filter 는 윈도우 크기의 값들에 대해서 정렬을 해야 하기 때문에 Min Filter 가 더 빠른 성능을 보여야 한다. 그러나, OpenCV 내부적으로 최적화가 잘 되어 있기 때문에, 직접 구현한 Min Filter 보다 월등한 연산 성능을 보인다.

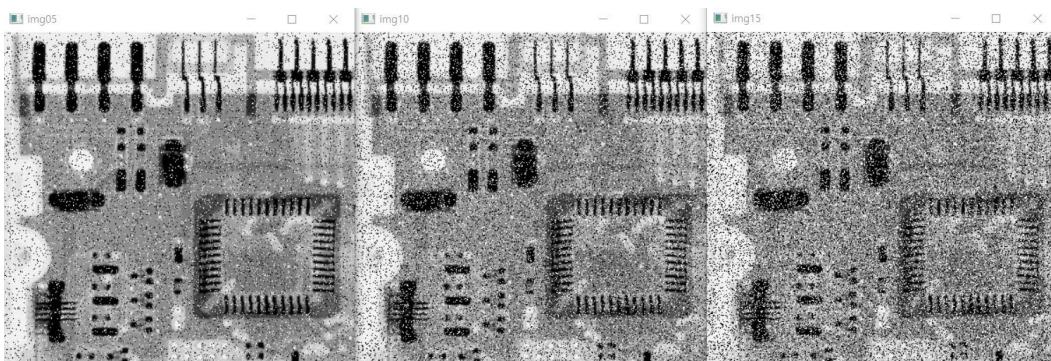
3-c. Salt Noise 제거 결론

시각적으로는 Min Filter 가 Salt Noise 제거를 잘 하는 것으로 보이나, Min Filter 는 전반적인 픽셀 값들이 최소값의 방향으로 가까워지는 경향을 띠기 때문에 이미지가 전체적으로 어두워진다. PSNR 수치만을 따지고 본다면, Median Filter 가 Min Filter 보다 훨씬 더 높은 값을 보인다. 그러나, 사람이 시각적으로 판단하기에는 Min Filter 가 훨씬 더 많은 Salt Noise 를 제거하기 때문에, 윈도우 크기가 3 인 Min Filter 가 Salt Noise 제거에 효과적이라는 결론을 내린다.

4. Salt And Pepper Noise

4-a. Median Filter

Salt And Pepper Noise 를 제거하기 위해서 Median Filter 와 Gaussian Filter 를 이용해서 필터링을 진행했다. 아래는 필터링을 적용하기 전의 이미지와 PSNR 값이다.

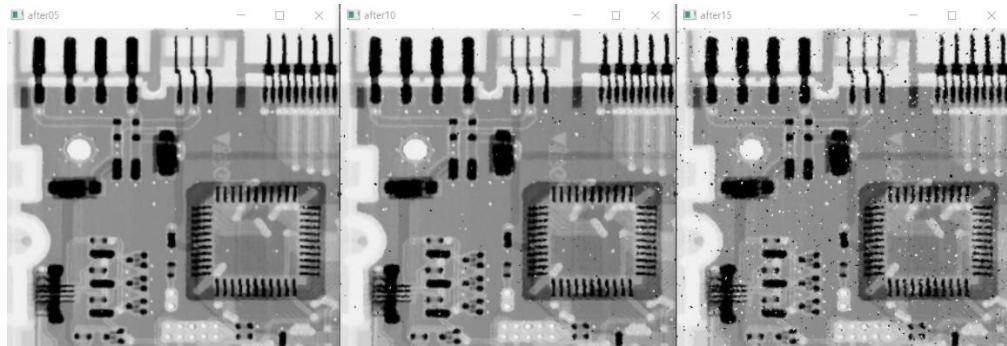


<Salt And Pepper Noise : 필터링 전>

Salt_Pepper	0.05	0.10	0.15
PSNR	63.1565	60.1453	58.3546

<Salt And Pepper Noise : PSNR>

육안으로 보기에도 이미지에 상당한 노이즈를 관찰할 수 있다. 위의 필터링 전의 이미지에 윈도 우크기를 바꿔가면서 Median Filter 를 적용해 봤다.



<Median Filtering : window size(3x3)>

필터링을 하기 전과 비교했을 때, 노이즈의 0.15 인 경우는 어느정도 노이즈가 관찰되긴 하지만, 상당히 많은 노이즈가 제거된 것을 확인할 수 있다.

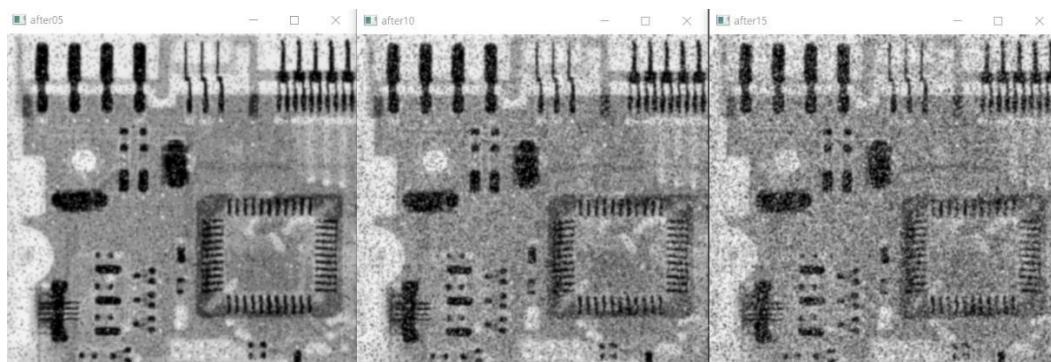
Median	0.05(PSNR)	0.10(PSNR)	0.15(PSNR)	소요시간(ms)
3x3	78.0713	74.2978	70.1689	0
5x5	74.36	73.4108	72.2248	1

<Median Filtering : PSNR>

원도우 크기가 3 일 때, 전반적으로 PSNR 이 증가한 것을 관찰할 수 있었는데, 3 보다 원도우 사이 즈가 커질수록 PSNR 수치가 점점 줄어드는 것을 확인할 수 있었다.

4-b. Gaussian Filter

Outlier 를 제거하는 다른 효과적인 방법은 주변의 픽셀 값들과 잘 녹아들게 섞는 것이다. Gaussian Filter 역시 Salt And Pepper Noise 제거에 효과적일 것이라고 생각해서 Gaussian Filtering 을 적용해 봤다.



<Gaussian Filter : 필터적용 후(시그마:1)>

Gaussian	0.05(PSNR)	0.10(PSNR)	0.15(PSNR)	소요시간(ms)
시그마 : 1	71.194	68.3637	66.2744	0
시그마 : 2	69.3109	67.9068	66.4599	1
시그마 : 5	65.4503	64.9404	64.3211	2

<Gaussian Filtering : PSNR>

시그마 값을 점진적으로 증가시키면서 관찰한 결과, 1 일 때가 시간도 적게 걸리고, PSNR의 수치도 노이즈 전반적으로 높게 나오는 것을 관찰했다. 시그마가 커질수록 PSNR의 수치는 점차 줄어드는 양상을 관찰할 수 있었다.

4-c. Salt And Pepper Noise 제거 결론

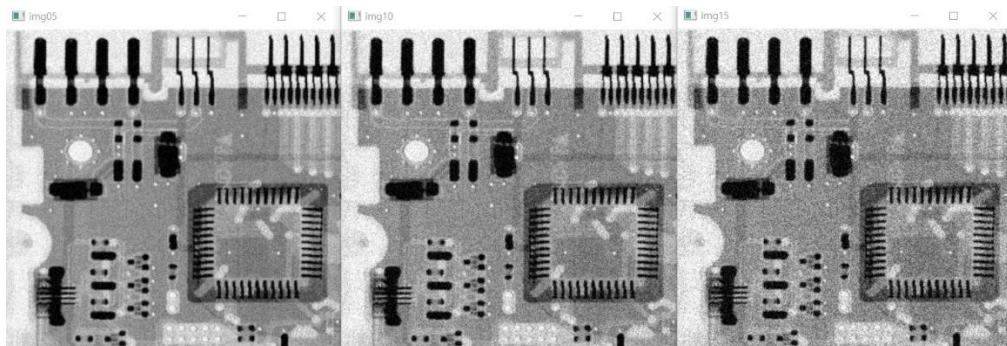
Median Filter 와 Gaussian Filter 모두 필터링을 하기 전과 비교하면, PSNR 수치가 증가하는 것을 볼 수 있었지만, Median Filter 가 더 효과적으로 Salt And Pepper Noise 를 제거할 수 있다는 사실을 시각적으로 수치적으로 확인이 가능했다.

Gaussian Filter 의 경우에는 주변의 픽셀과의 연산을 통해서 원본보다 이미지를 부드럽게 만들기는 하지만, 자기 자신의 가중치가 가장 높다. 노이즈가 있는 픽셀 부분을 필터링 한다고 했을 때, 노이즈 값의 영향력이 제일 클 수밖에 없다. 따라서 시각적으로 노이즈 픽셀의 흔적이 남아 있을 확률이 크다. Median Filter 의 경우에는 윈도우에서 중앙값을 사용하므로 Salt And Pepper Noise 제거에 윈도우 사이즈가 3 인 Median Filter 가 효과적이다.

5. Uniform Noise

5-a. Mean Filter

Uniform Noise 를 제거하기 위해서 Mean Filter 와 직접 구현한 Midpoint Filter 를 사용했다. 우선, 필터링 적용 전의 이미지와 PSNR 값이다.

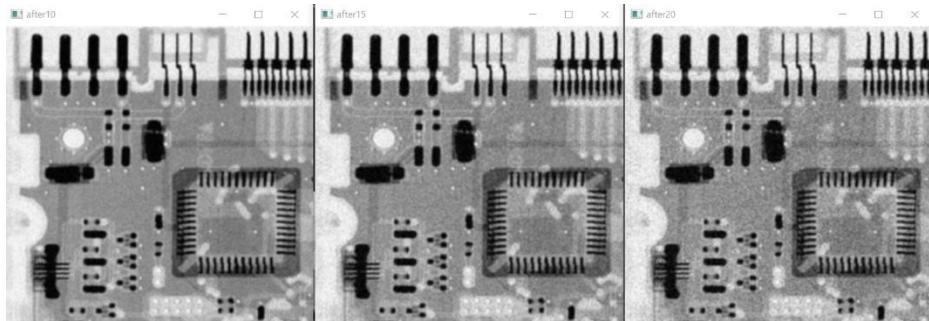


< Uniform Noise : 필터링 적용 전 >

Uniform	0.10	0.15	0.20
PSNR	73.1277	69.7173	67.3135

<Uniform Noise : PSNR>

Mean Filter, 정확하게는 Arithmetic Filter 의 윈도우 사이즈를 점차 증가시키면서 관찰해본 결과는 아래와 같다.



<Arithmetic Filtering : window size(3x3)>

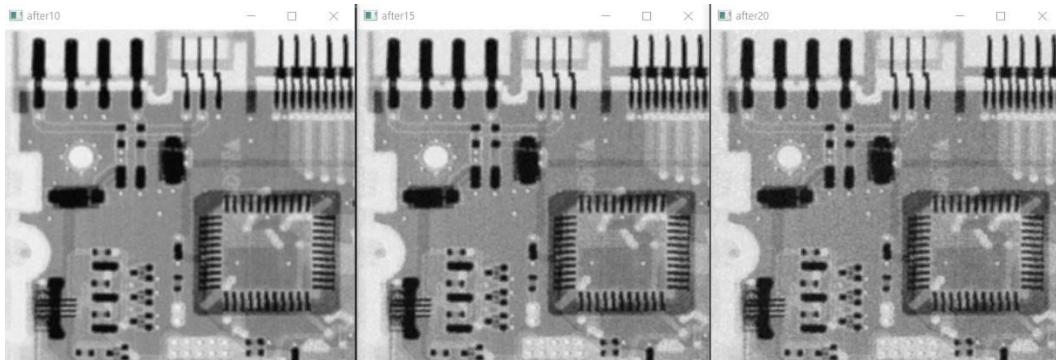
Median	0.05(PSNR)	0.10(PSNR)	0.15(PSNR)	소요시간(ms)
3x3	78.0713	74.2978	70.1689	4
5x5	72.0360	71.7254	71.3272	7

<Arithmetic Filtering : PSNR>

다른 Noise 의 결과와 동일하게 윈도우 크기가 3 일 때가 PSNR 수치가 가장 높게 나오는 것을 확인 할 수 있었다. 역시 다른 관찰들과 동일하게 윈도우 크기가 커질수록 PSNR 의 수치가 감소하고, 시간도 오래 걸린다는 것을 확인할 수 있었다. 시각적으로 볼 때는 필터링 전후의 차이는 느낄 수 없었다.

5-b. MidPoint Filter

MidPoint Filter 는 윈도우에서 최대값과 최소값의 합의 평균을 픽셀의 값으로 사용하는 필터인데, Gaussian Noise 나 Uniform Noise 와 같이 무작위로 분포해 있는 noise 를 제거하는데 효과적이다. 윈도우 사이즈를 변경하면서 관찰했다.



<MidPoint Filtering : window size(3x3)>

MidPoint	0.10(PSNR)	0.15(PSNR)	0.20(PSNR)	소요시간(ms)
3x3	73.8716	73.0775	72.0789	3
5x5	68.1686	67.9763	67.6449	7

<Midpoint Filtering : PSNR>

윈도우 크기가 3 인 경우, 노이즈가 0.10 일 때 PSNR 의 유의미한 수치의 변화는 관찰할 수 없었다. 노이즈가 정도가 커진 0.15, 0.20 에서는 PSNR 의 수치가 약간 상승하는 것을 볼 수 있었다.

5-c. Uniform Noise 제거 결론

Mean Filter 와 MidPoint Filter 를 이용해서 필터링을 진행했는데, Mean Filter 가 전반적으로 높은 PSNR 수치를 보여줬다. MidPoint Filter 는 Max 와 Min 의 평균을 통해서 값을 결정하는데, 만약 원 도우가 저주파와 고주파 영역에 걸쳐서 있다면, 의도와는 다른 왜곡된 값을 얻을 가능성이 크다. Mean Filter 역시도 밝기의 변화의 기울기가 변하는 구간에서는 효과를 보기 힘든데, 위의 이미지 에서는 Mean Filter 가 MidPoint Filter 에 비해서 상대적으로 덜 왜곡된 값을 얻었기 때문에 Mean 필터링 방식이 더 좋은 결과를 얻지 않았나 추측한다.