5-1. 영상 노이즈 유형

잡음(noise)

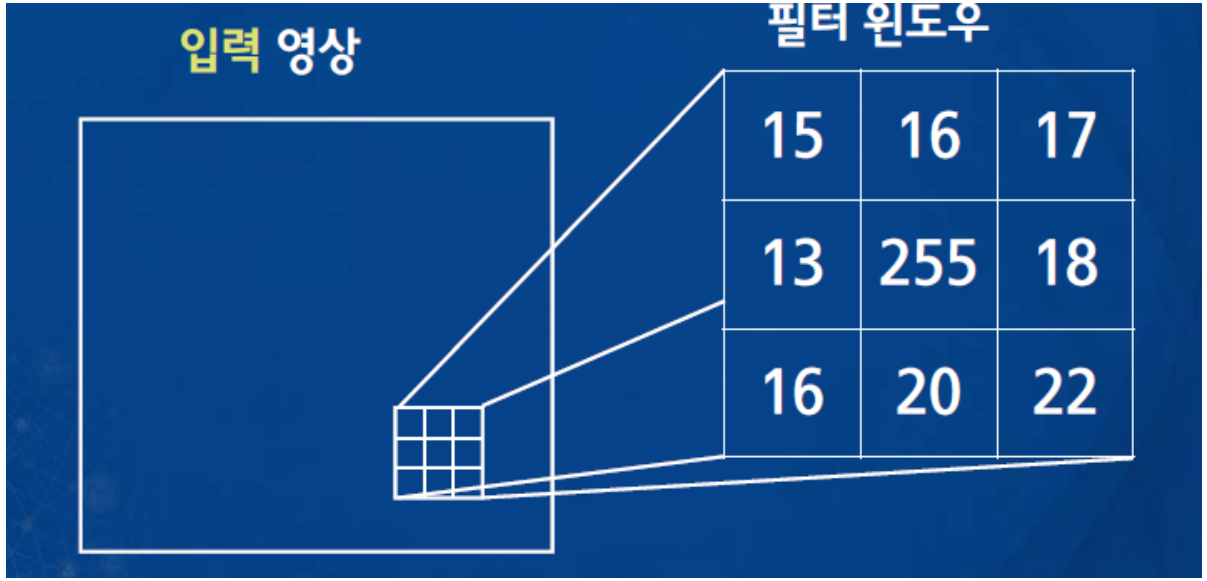
- 화소 값이 특별하게 바뀌어 있는 것
- 가우시안 잡음 : 영상 화소값이랑 비슷한데 불규칙한 잡음
- 임펄스 잡음 : 영상 화소값과 뚜렷하게 다른 잡음

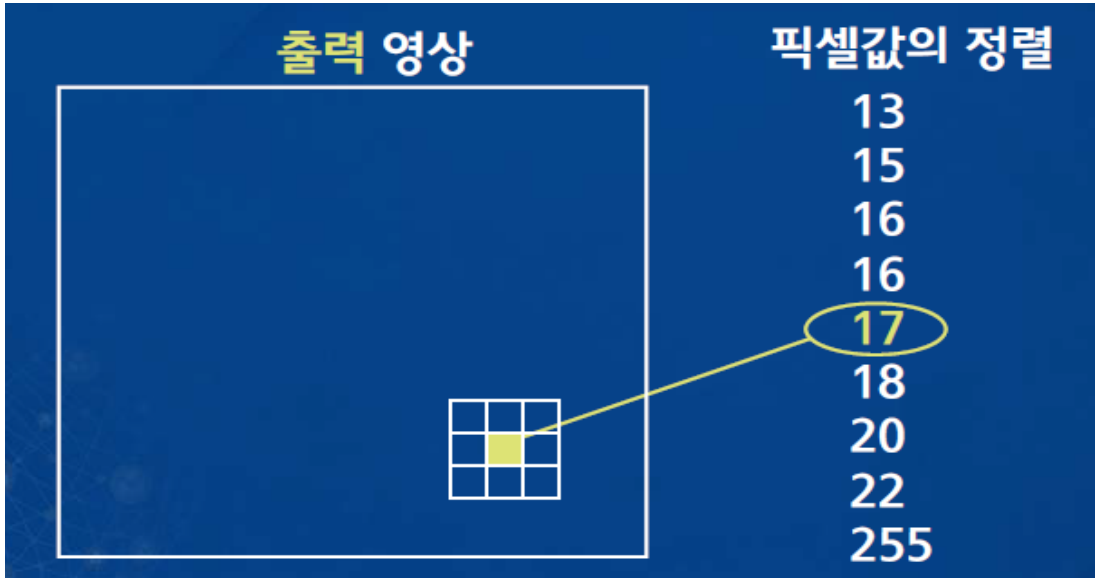
잡음 제거 - 평균 필터 마스크(Convolution)

- 잡음은 흐려지지만 퍼지는 단점이 있다.
- 임펄스 잡음에는 비효과적(노이즈가 더 커짐)

잡음 제거 – 중간값 필터링

- 픽셀을 정렬하여 중간에 있는 값을 Center position에 위치시키는 것
- 임펄스 잡음에 효과적이다.
- 노이즈 : 3x3 영역에서 1개 이하





5-2. 영상 노이즈 제거

임펄스 노이즈 제거

- Salt & Pepper 노이즈라고도 부름(0 또는 1, 흰색 혹은 검정색)

소스코드

```
void main()
{
    BITMAPFILEHEADER hf; // 14bytes
    BITMAPINFOHEADER hInfo; // 40bytes
    RGBQUAD hRGB[256]; // 1024bytes
    FILE * fp;
    fp = fopen("salt.bmp", "rb");
    if (fp == NULL){
        printf("File not foud₩n");
        return; // 프로그램 종료
    }
```

이미지를 읽어온다.

```
fread(&hf, sizeof(BITMAPFILEHEADER), 1, fp);
fread(&hInfo, sizeof(BITMAPINFOHEADER), 1, fp);
fread(hRGB, sizeof(RGBQUAD), 256, fp);
int W, H, ImgSize;
W = hInfo.biWidth;
H = hInfo.biHeight;
ImgSize = W*H;
BYTE * Image = (BYTE *)malloc(ImgSize);
BYTE * Output = (BYTE *)malloc(ImgSize);
fread(Image, sizeof(BYTE), ImgSize, fp);
fclose(fp);
```

이미지 값들을 할당한다.

소스코드

```
// 영상처리
BYTE Temp[9];  화소값 저장 배열
for (int i = 1; i < H - 1; i++){   필터(커널) Height  : 위->아래
    for (int j = 1; j < W - 1; j++){ 필터(커널) Width  : 좌->우
        for (int m = -1; m <= 1; m++){ 필터(커널) 중심  : 위->아래
            for (int n = -1; n <= 1; n++){필터(커널) 중심  : 좌->우
                Temp[(m+1)*3+(n+1)] = Image[(i+m)*W + (j+n)]; 화소값 저장
            }
        }
        Sorting(Temp, 9);                  정렬 후 Output 배열에
        Output[i*W + j] = Temp[4];         수정된 중심 값들 적용
    }
}
```

소스코드

```
void Sorting(BYTE * Arr, int Size) // 오름차순 정렬
{
    for (int i = 0; i < Size - 1; i++) // Pivot Index
    {
        for (int j = i + 1; j < Size; j++) // 비교대상 Index
        {
            if (Arr[i] > Arr[j]) // 비교대상이 더 작다면...
                swap(&Arr[i], &Arr[j]);
        }
    }
}
```

```
void swap(BYTE* a, BYTE* b)
{
    BYTE temp = *a;
    *a = *b;
    *b = temp;
}
```

버블정렬