12장 필터링을 이용한 영상처리

- 필터링의 개념
- 선형 공간 영역 필터링
- 선형 공간 필터링을 이용한 잡음제거
- 비선형 공간 필터링을 이용한 잡음제거

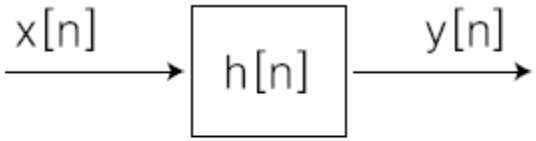
한빛미디어(주)

학습목표

- ✓ 선형 공간 필터링의 동작을 소개하고 응용을 공부한다.
- ✓ 평균 필터를 이용한 잡음제거를 소개한다.
- ✓ 중간 값 필터와 최소/최대 필터를 이용한 잡음제거를 소개한다.

Section 01 필터링의 개념

- 필터
  - 입력되는 신호의 일부 성분을 제거하거나 일부 특성을 변경하려고 설계된 하나의 시스템



[그림 12-1] 시스템의 개념

- 필터 종류
  - 유한임펄스응답(**Finite Impulse Response: FIR**) 필터
    - 필터의 길이가 한정된 필터
    - 설계가 쉽고, 신호도 쉽게 처리할 수 있음.
  - 무한임펄스응답(**Infinite Impulse Response: IIR**) 필터
    - 필터의 길이가 무한정한 필터
    - 설계가 어렵고 이를 처리도 힘들나, 필터의 특성은 더 우수
    - 영상처리에서는 효과적인 필터링의 특성을 만족하면서 선형 시불변 시스템의 특성도 만족하는 **FIR** 필터를 많이 사용함.

컨벌루션

- 임의의 디지털 신호 x[n]이 선형 시불변 시스템인 FIR 필터에 입력되어 원하는 출력 y[n]을 만드는 과정

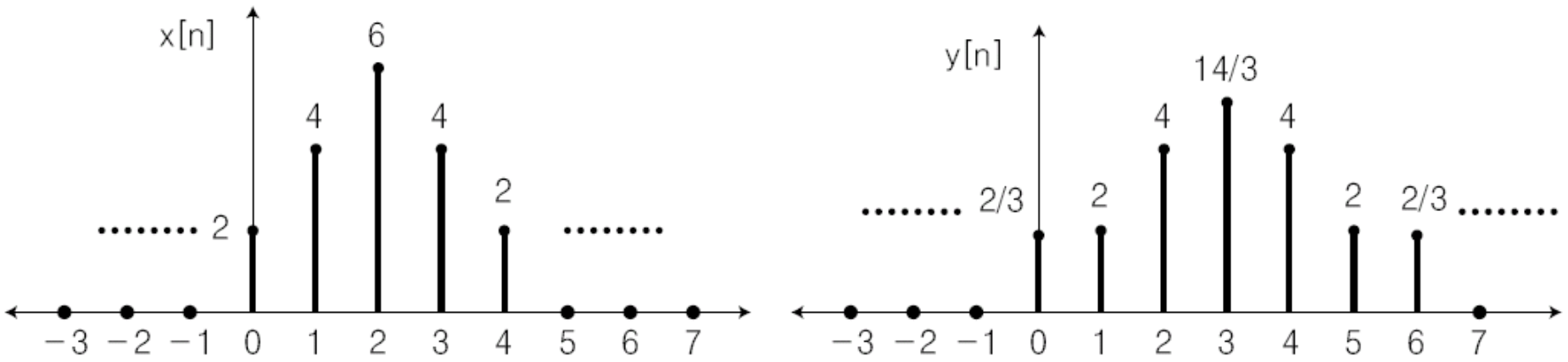
  - 값 3개의 평균을 구하는 입출력 관계식

$$y[n] = \frac{1}{3}(x[n] + x[n-1] + x[n-2])$$

  - [표12-1] 주어진 입력 x[n]에서 출력 y[n]을 구하는 과정

n	···	−2	−1	0	1	2	3	4	5	6	7	···
x[n]	0	0	0	2	4	6	4	2	0	0	0	0
y[n]	0	0	0	2/3	2	4	14/3	4	2	2/3	0	0

컨벌루션(계속)



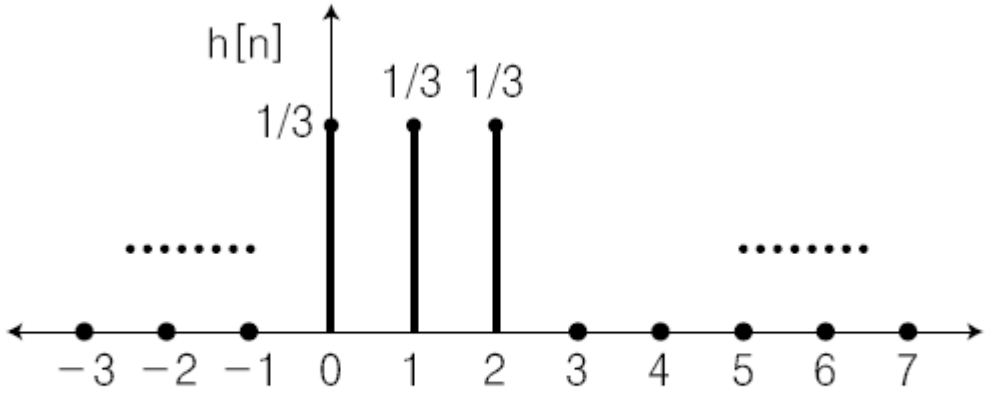
[그림 12-2] 입력 x[n]과 출력 y[n]

- 출력을 얻으려고 사용한 선형 시불변 시스템 **FIR** 필터

$$h[n]=\frac{1}{3}\delta[n]+\frac{1}{3}\delta[n-1]+\frac{1}{3}\delta[n-2]$$

  - 여기서 $\delta[n]=\begin{cases}1, & n=0\\ 0, & n\neq 0\end{cases}$ 으로 정의

컨벌루션(계속)



[그림 12-3] FIR 필터 h[n]

- 입력 x[n]과 FIR 필터 h[n]과의 관계에서 다음과 같은 출력 y[n]

$$y[n] = \sum_{k=0}^{M} h[k]x[n-k]$$

  - M은 필터의 길이, 현재 사용되는 필터의 길이는 2인 컨벌루션
  - 선형 시불변 시스템에 입력되는 신호가 어떤 신호를 출력하는지 결정해줌.

회선 처리를 이용한 영상의 필터링

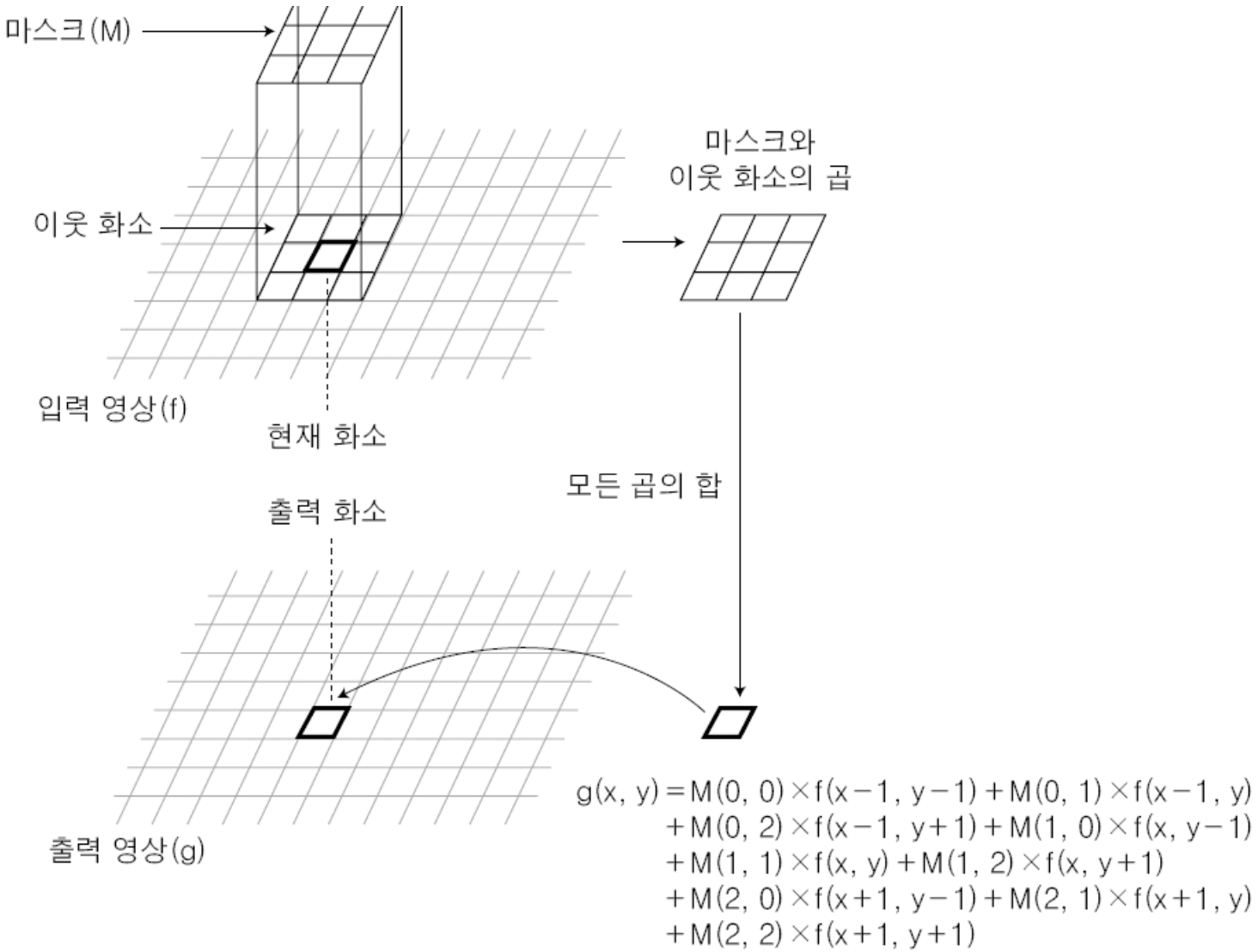
- 필터링을 이용한 영상처리는 2차원의 컨벌루션을 수행하게 됨.
  - 영상의 공간 필터링은 크기가 M×N인 FIR 필터 마스크 h[x, y]와 크기가 M×N인 영상 간에 2차원 컨벌루션을 수행하는 것

$$y[x,y] = \sum_{k=0}^{M} \sum_{l=0}^{N} h[k,l]x[x-k, y-l]$$

    - 사용되는 필터 마스크를 컨벌루션 마스크 또는 회선 마스크라고 함.
- N×N 회선 마스크는 폭이 N이고 서로 직교하는 1차원 마스크 두 개를 곱하여 생성
  - FIR 필터의 계수가 [1 -2 1]이라고 가정하면, 다음과 같이 3×3의 2차원 회선 마스크 생성 가능

$$\begin{bmatrix} 1 & 0 & 0 \\ -2 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & -2 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 1 & -2 & 1 \\ -2 & 4 & -2 \\ 1 & -2 & 1 \end{bmatrix}$$

회선 처리를 이용한 영상의 필터링(계속)



[그림 12-4] 영상처리를 위한 2차원 컨벌루션의 동작 과정

- 공간 필터링(Spatial Filtering)
  - 영상에 있는 공간 주파수 대역을 제거하거나 강조하는 필터 처리
  - 사용되는 필터의 계수 따라 특정 주파수를 제거하거나 강조하므로, 필터 마스크 또는 회선 마스크의 가중치 선택이 공간 필터의 행동을 결정
  - 영상처리에서는 홀수 차원의 정방형 마스크가 사용됨.

a	b	c
d	e	f
g	h	i

[그림 12-5] 2차원 필터의 계수 또는 회선 마스크

- 공간 필터링 연산의 분류
  - 저주파 통과 필터링
    - 저주파 성분을 남기고 고주파 성분을 제거하는 필터링
  - 고주파 통과 필터링
    - 고주파 성분을 남기고 저주파 성분을 제거하는 필터링
  - 에지 강화 필터
    - 경계선 검출

저주파 통과 필터링(Low-Pass Filter: LPF)

- 신호 성분 중 저주파 성분은 통과시키고 고주파 성분은 차단하는 필터
- 잡음을 제거하거나 흐릿한 영상을 얻을 때 주로 사용되는 필터
- 고주파 성분을 제거하므로 고주파 차단 필터라고도 함.

$$\frac{1}{9}\times\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix} \quad \frac{1}{12}\times\begin{bmatrix}1 & 1 & 1\\ 1 & 4 & 1\\ 1 & 1 & 1\end{bmatrix} \quad \frac{1}{18}\times\begin{bmatrix}1 & 1 & 1\\ 1 & 10 & 1\\ 1 & 1 & 1\end{bmatrix}$$

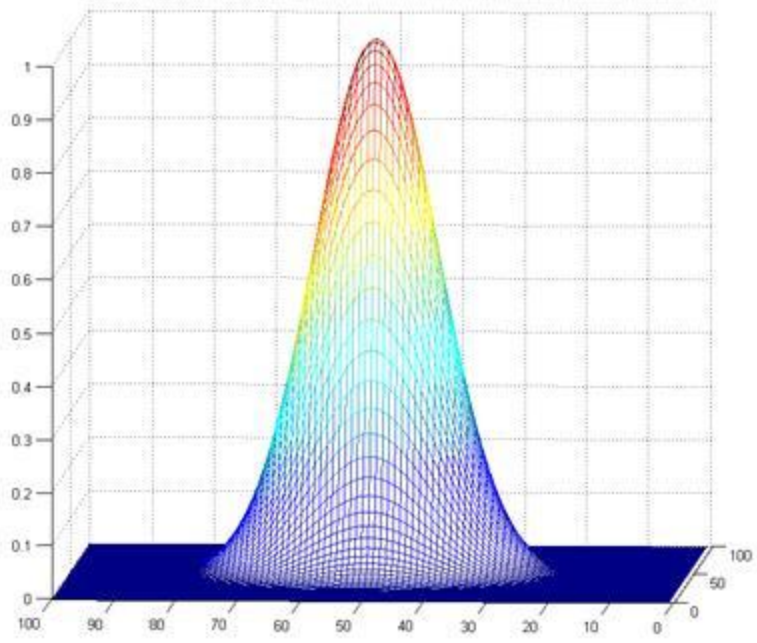
[그림 12-6] 저주파 통과 필터 마스크

  - 저주파 통과 필터링의 마스크는 모든 계수가 양수이고 전체 합이 1인 마스크가 사용됨

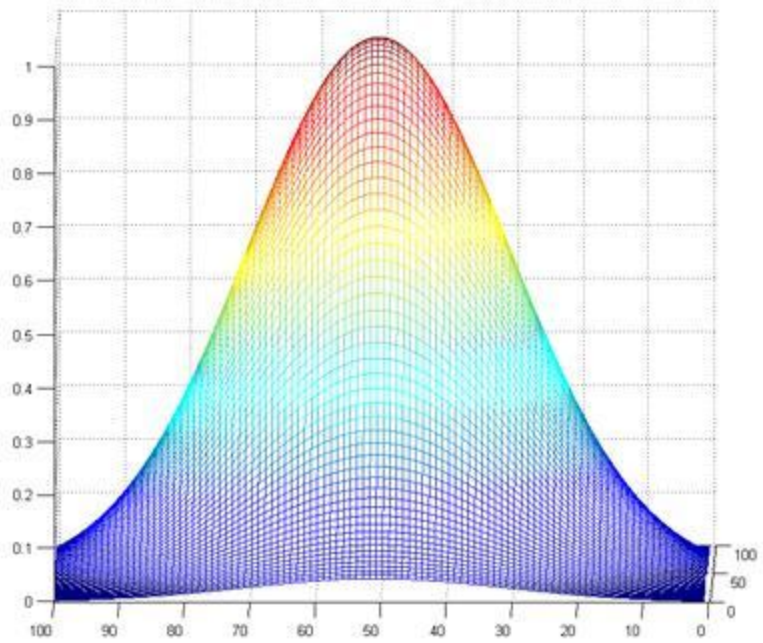
- 가우시안 필터는 가우시안 함수를 표본화하여 마스크의 계수를 결정

$$h(x,y)=\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)$$

저주파 통과 필터링(계속)



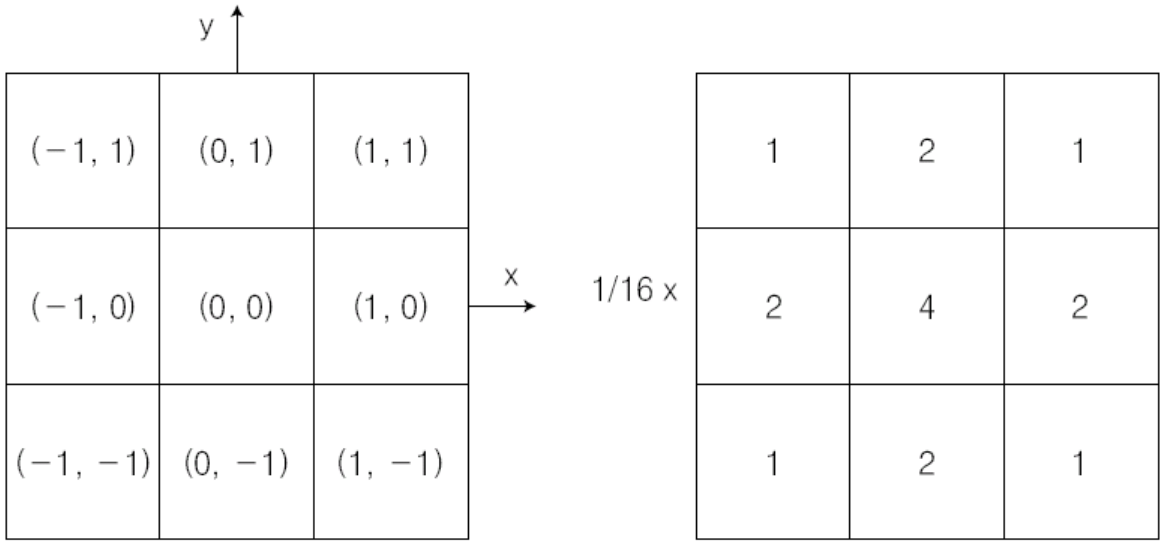
$\delta = 4$



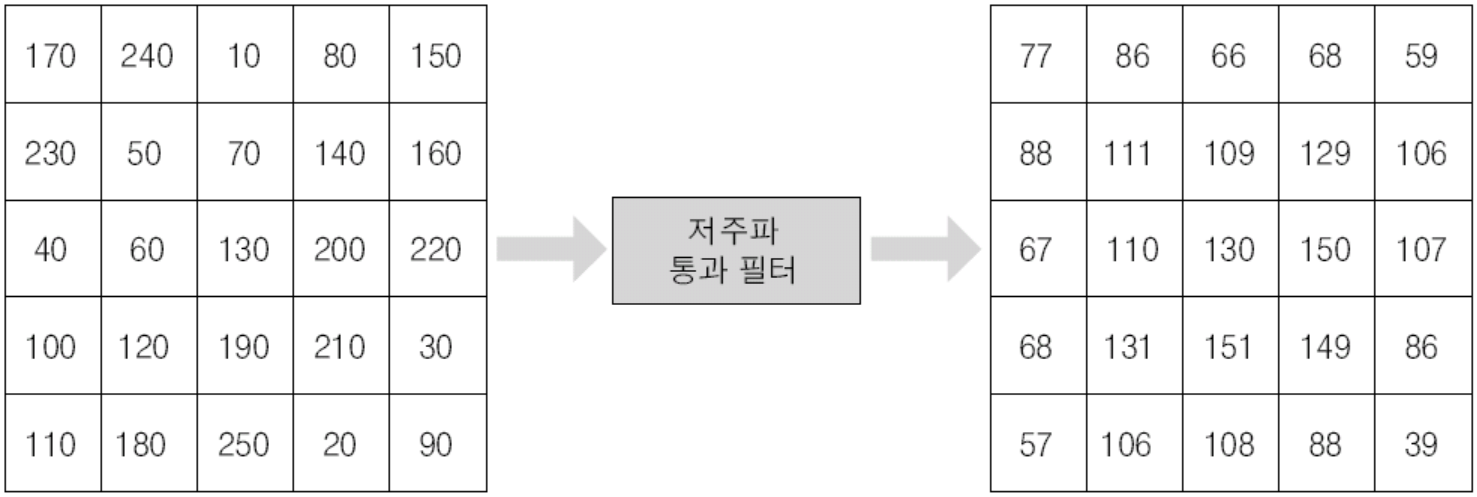
$\delta = 9$

[그림 12-7] 2차원 가우시안 함수

저주파 통과 필터링(계속)



[그림 12-8] 가우시안 마스크



(a) 입력 영상 (b) 출력 영상

[그림 12-9] 저주파 통과 필터링을 이용한 화소 값의 변화

[실습하기 12-1] 저주파 통과 필터링 프로그램

① **ResourceView 창에서 [Menu]-[IDR_IMAGETYPE] 더블클릭 → 메뉴 추가**

ID	ID_LOW_PASS_FILTER
Caption	저주파 통과 필터링

② **[MFC ClassWizard] 대화상자를 이용해 추가된 메뉴에서 저주파 통과 필터링을 실행하는 함수 추가**

Class Name	Function Type	Function Name
View Class	void	OnLowPassFilter
Doc Class	void	OnLowPassFilter

③ **Doc 클래스에 다음 프로그램 추가**

[실습하기 12-1] 저주파 통과 필터링 프로그램

```
void CImageProcessingDoc::OnLowPassFilter()
{
   int i, j;
   double LPF[3][3]  = {{1./9., 1./9., 1./9.},
                        {1./9., 1./9., 1./9.},
                        {1./9., 1./9., 1./9.}};
   //double LPF[3][3] = {{1./12.,1./12.,1./12.},
                         {1./12.,4./12.,1./12.},
                         {1./12.,1./12.,1./12.}};
   //double LPF[3][3] = {{1./18.,1./18.,1./18.},
                         {1./18.,10./18.,1./18.},
                         {1./18.,1./18.,1./18.}};
   // 저주파 필터 마스크
   m_Re_height = m_height;
   m_Re_width = m_width;
   m_Re_size = m_Re_height * m_Re_width;

   m_OutputImage = new unsigned char [m_Re_size];

   m_tempImage = OnMaskProcess(m_InputImage, LPF);
   // 입력 영상과 마스크를 이용한 회선 처리
```

[실습하기 12-1] 저주파 통과 필터링 프로그램

```
 for(i=0 ; i<   m_Re_height ; i++){
      for(j=0 ; j<   m_Re_width ; j++){
         if(m_tempImage[i][j] > 255)
         // 회선 처리 결과 값이 0~255 사이의 값이 아닐 때
         // 0보다 작으면 0을,
         // 255보다 크면 255를 출력
            m_OutputImage[i*   m_Re_width + j] = 255;
         else if(m_tempImage[i][j] < 0)
            m_OutputImage[i*   m_Re_width + j] = 0;
         else
            m_OutputImage[i*   m_Re_width + j]
               = (unsigned char)m_tempImage[i][j];
      }
   }
}
```

[실습하기 12-1] 저주파 통과 필터링 프로그램

④ View 클래스에 다음 프로그램 추가

```
void CImageProcessingView::OnLowPassFilter()
{
    CImageProcessingDoc* pDoc = GetDocument();
    ASSERT_VALID(pDoc);

    pDoc->OnLowPassFilter();

    Invalidate(TRUE);
}
```

[실습하기 12-1] 저주파 통과 필터링 프로그램

⑤ 프로그램 실행 결과 영상

- 블러링에서 보았던 바와 같이 영상이 전체적으로 흐려짐.

(a) 원본 영상 (b) 저주파 통과 영상 1 (c) 저주파 통과 영상 2

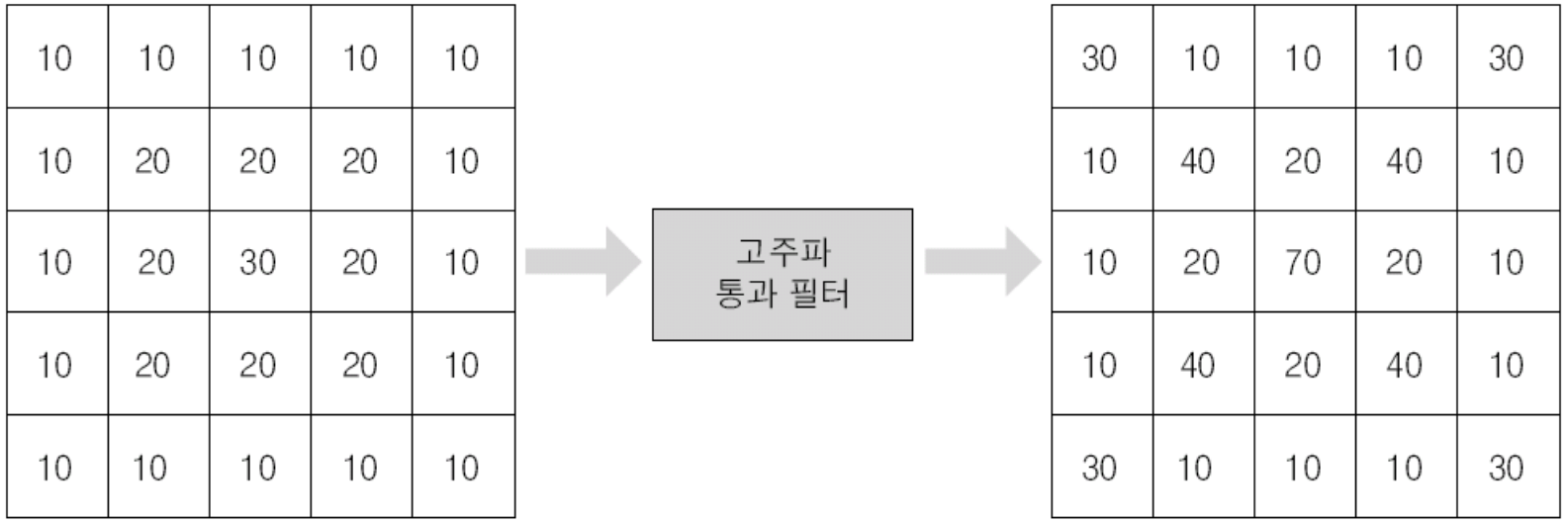
실제 영상에서 여러 저주파 통과 필터링을 수행한 결과 영상

고주파 통과 필터링 (High-Pass Filter: HPF)

- 신호 성분 중 고주파 성분은 통과시키고 저주파 성분은 차단하는 필터
- 저주파 성분을 차단하므로 저주파 차단 필터라고도 함.
- 고주파 통과 필터링은 영역 처리에서 배운 샤프닝(Sharpening)과 같은 처리 방법
- 흐려진 영상을 개선하여서 첨예화하는 결과 영상 생성

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

고주파 통과 필터링(계속)



(a) 입력 영상 (b) 결과 영상

[그림 12-11] 고주파 통과 필터링을 이용한 화소 값의 변화

- 고주파 통과 필터 영상은 저주파 통과 필터를 이용하여 얻을 수 있음.
- 원본 영상에서 저주파 통과 필터링으로 얻은 영상 뺀 차 영상은 고주파 성분만 남게 됨.

$$f_H(x,y) = f(x,y) - f_L(x,y)$$

- $f_H(x, y)$: 고주파 영상, $f(x, y)$: 원본 영상, $f_L(x, y)$: 저주파 영상

고주파 통과 필터링(계속)

- 고주파 강조 필터
  - 고주파에 해당하는 세부 정보를 강조하는 반면, 영상에서 중요한 부분인 낮은 공간 주파수의 성분은 손실시키는 고주파 통과 필터의 문제 해결
  - 저주파 영역의 상쇄에 해당하는 부분에 일정량의 이득을 주어 낮은 공간 주파수에 해당하는 성분의 손실을 어느 정도 보상할 수 있음.
- 고주파 강조 필터 생성 방법

$$g(x,y) = Af(x,y) - f_L(x,y) = (A-1)f(x,y) + f(x,y) - f_L(x,y)$$
$$= (A-1)f(x,y) + f_H(x,y)$$

- 고주파 강조 필터 마스크

$$\frac{1}{9} \times \begin{bmatrix} -1 & -1 & -1 \\ -1 & \alpha & -1 \\ -1 & -1 & -1 \end{bmatrix} \quad \alpha = 9A - 1$$

[그림 12-12] 고주파 강조 필터 마스크

고주파 통과 필터링(계속)

- 샤프닝 필터는 고주파 통과 필터에서 발생하는 낮은 공간 주파수의 성분이 손실되는 문제점을 보완해 주는 회선 마스크
- 샤프닝 필터링된 영상은 원본 영상에 고주파 통과 필터링된 영상을 합한 것과 비슷한 결과를 얻음.

$$\begin{bmatrix} 1 & -2 & 1 \\ -2 & 5 & -2 \\ 1 & -2 & 1 \end{bmatrix} \quad \begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix} \quad \begin{bmatrix} -1 & -1 & -1 \\ -1 & 0 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

[그림 12-13] 샤프닝 필터 마스크

[실습하기 12-2] 고주파 통과 필터링 프로그램

① **ResourceView 창에서 [Menu]-[IDR_IMAGETYPE] 더블클릭 → 메뉴 추가**

ID	ID_HIGH_PASS_FILTER
Caption	고주파 통과 필터링

② **[MFC ClassWizard] 대화상자를 이용해 추가된 메뉴에서 고주파 통과 필터링을 실행하는 함수 추가**

Class Name	Function Type	Function Name
View Class	void	OnHighPassFilter
Doc Class	void	OnHighPassFilter

③ **Doc 클래스에 다음 프로그램 추가**

[실습하기 12-2] 고주파 통과 필터링 프로그램

```
void CImageProcessingDoc::OnHighPassFilter()
{
   int i, j;
   double HPF[3][3]  = {{-1./9., -1./9., -1./9.},
                        {-1./9., 8/9., -1./9.},
                        {-1./9., -1./9., -1./9.}};
   //double HPF[3][3] = {{-1., -1., -1.},
                        {-1., 9., -1.},
                        {-1., -1., -1.}};
   // 고주파 필터 마스크
   m_Re_height = m_height;
   m_Re_width = m_width;
   m_Re_size = m_Re_height * m_Re_width;

   m_OutputImage = new unsigned char [m_Re_size];

   m_tempImage = OnMaskProcess(m_InputImage, HPF);
```

[실습하기 12-2] 고주파 통과 필터링 프로그램

```
for(i=0 ; i<   m_Re_height ; i++){
      for(j=0 ; j<   m_Re_width ; j++){
         // 회선 처리 결과 값이 0~255 사이의 값이 아닐 때
         // 0보다 작으면 0을,
         // 255보다 크면 255를 출력
         if(m_tempImage[i][j] > 255)
            m_OutputImage[i*   m_Re_width + j] = 255;
         else if(m_tempImage[i][j] < 0)
            m_OutputImage[i*   m_Re_width + j] = 0;
         else
            m_OutputImage[i*   m_Re_width + j]
               = (unsigned char)m_tempImage[i][j];
      }
   }
}
```

[실습하기 12-2] 고주파 통과 필터링 프로그램

④ View 클래스에 다음 프로그램 추가

```
void CImageProcessingView::OnHighPassFilter()
{
    CImageProcessingDoc* pDoc = GetDocument();
    ASSERT_VALID(pDoc);

    pDoc->OnHighPassFilter();

    Invalidate(TRUE);
}
```

[실습하기 12-2] 고주파 통과 필터링 프로그램

⑤ 프로그램 실행 결과 영상

- (b)는 낮은 주파수 성분이 많이 제거되어 경계선이 확연하게 보임.
- (c)~(f)는 영상에서 중요 성분은 그대로 남아 있는 채 경계선 부분이 강조됨.

(a) 원본 영상

(b) 고주파 통과

(c) 고주파 강조($\alpha = 17$)

고주파 성분을 필터링 처리한 결과 영상

(d) 샤프닝 1

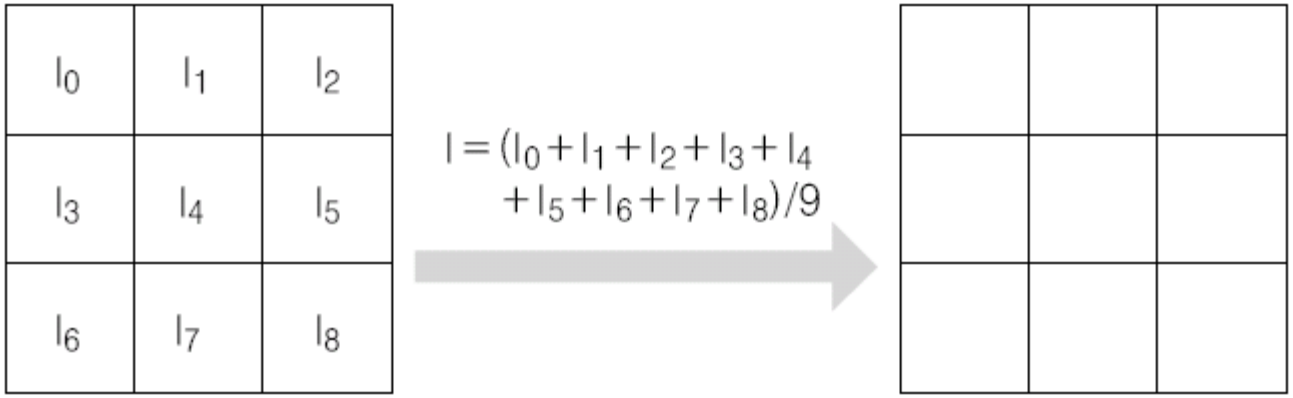
(e) 샤프닝 2

(f) 샤프닝 3

Section 03 선형 공간 필터링을 이용한 잡음제거

- 선형 공간 필터링을 이용한 잡음제거 기법
  - 저주파 통과 필터를 이용하는 방법
  - 회선 마스크의 계수와 곱한 화소의 선형 합으로 연산 수행
  - 저주파 통과 필터를 평균 필터라고도 함.

- 저주파 통과 필터의 동작
  - 영상을 흐리게 하는 블러링 처리
  - 주변 화소를 평균하므로 저주파 통과 필터가 영상을 흐리게 할 수 있음.
  - 저주파 통과 필터를 평균 필터라고도 함.



[그림 12-14] 평균 필터의 원리

[실습하기 12-3] 평균 필터링으로 잡음 제거하는 프로그램

① **ResourceView 창에서 [Menu]-[IDR_IMAGETYPE] 더블클릭 → 메뉴 추가**

ID	ID_MEAN_FILTER
Caption	평균 필터링

② **[MFC ClassWizard] 대화상자를 이용해 추가된 메뉴에서 평균 필터링을 실행하는 함수 추가**

Class Name	Function Type	Function Name
View Class	void	OnMeanFilter
Doc Class	void	OnMeanFilter

③ **Doc 클래스에 다음 프로그램 추가**

[실습하기 12-3] 평균 필터링으로 잡음 제거하는 프로그램

```
void CImageProcessingDoc::OnMeanFilter()
{
   int i, j, n, m;

   m_Re_height = m_height;
   m_Re_width = m_width;
   m_Re_size = m_Re_height * m_Re_width;

   m_OutputImage = new unsigned char [m_Re_size];

   double **tempInputImage, **tempOutputImage, S = 0.0;

   tempInputImage = Image2DMem(m_height + 2, m_width + 2);
   tempOutputImage = Image2DMem(m_height, m_width);

   for(i=0 ; i<m_height ; i++){
      for(j=0 ; j<m_width ; j++){
         tempInputImage[i+1][j+1]
            = (double)m_InputImage[i * m_width + j];
      }
   }
```

[실습하기 12-3] 평균 필터링으로 잡음 제거하는 프로그램

```
for(i=0 ; i<m_height ; i++){
      for(j=0 ; j<m_width ; j++){
         for(n=0 ; n<3 ; n++){
            for(m=0 ; m<3 ; m++){
               S += tempInputImage[i+n][j+m];
               // 입력 영상에서 3*3 크기의 배열 값을 누적
            }
         }
         m_OutputImage[i*   m_Re_width + j]
            = (unsigned char)(S/9.); // 평균값 출력
         S = 0.0; // reset
      }
   }
}
```

[실습하기 12-3] 평균 필터링으로 잡음 제거하는 프로그램

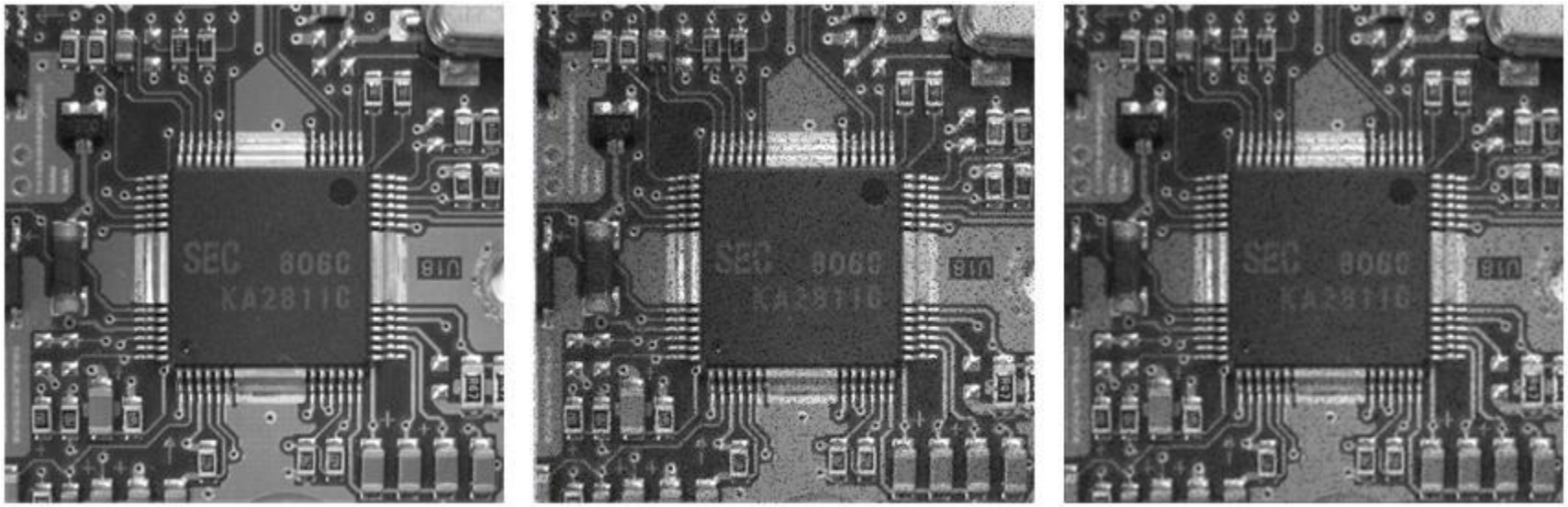
④ View 클래스에 다음 프로그램 추가

```
void CImageProcessingView::OnMeanFilter()
{
    CImageProcessingDoc* pDoc = GetDocument();
    ASSERT_VALID(pDoc);

    pDoc->OnMeanFilter();

    Invalidate(TRUE);
}
```

[실습하기 12-3] 평균 필터링으로 잡음 제거하는 프로그램

⑤ 프로그램 실행 결과 영상

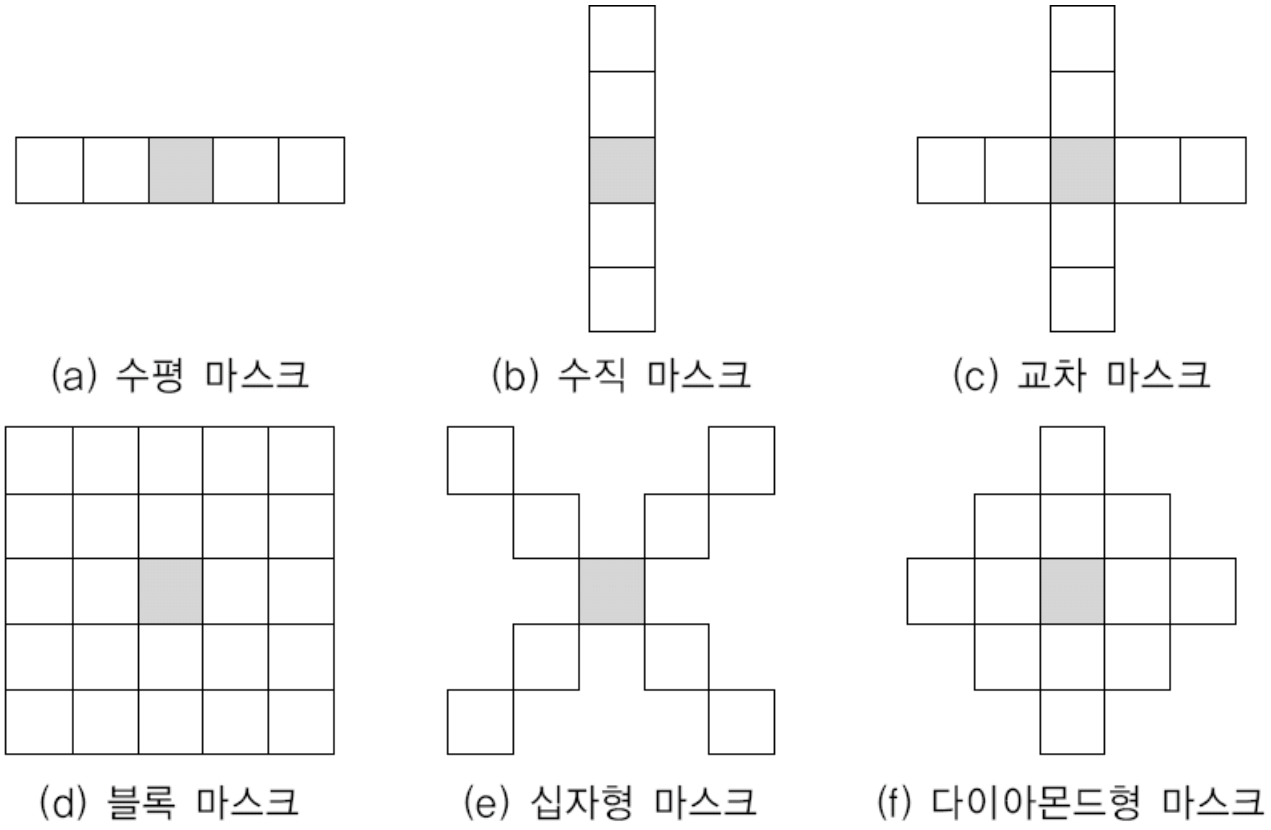
- (a)는 입력된 원본 영상
- (b)는 입력 영상에 임펄스 잡음을 첨가한 것
- (c)는 평균 필터로 임펄스 잡음의 제거를 시도한 결과 영상



(a) 원본 영상 (b) 잡음 첨가 (c) 평균 필터링

평균 필터로 잡음제거한 영상

- 중간 값 필터링으로 잡음제거
  - 중간 값 필터(Median Filter, 미디언 필터)는 이웃 화소의 값을 오름차순으로 정렬한 뒤 가운데에 있는 값을 출력 값으로 선택
  - 제거하려는 잡음에 따라 중간 값 필터의 마스크도 결정

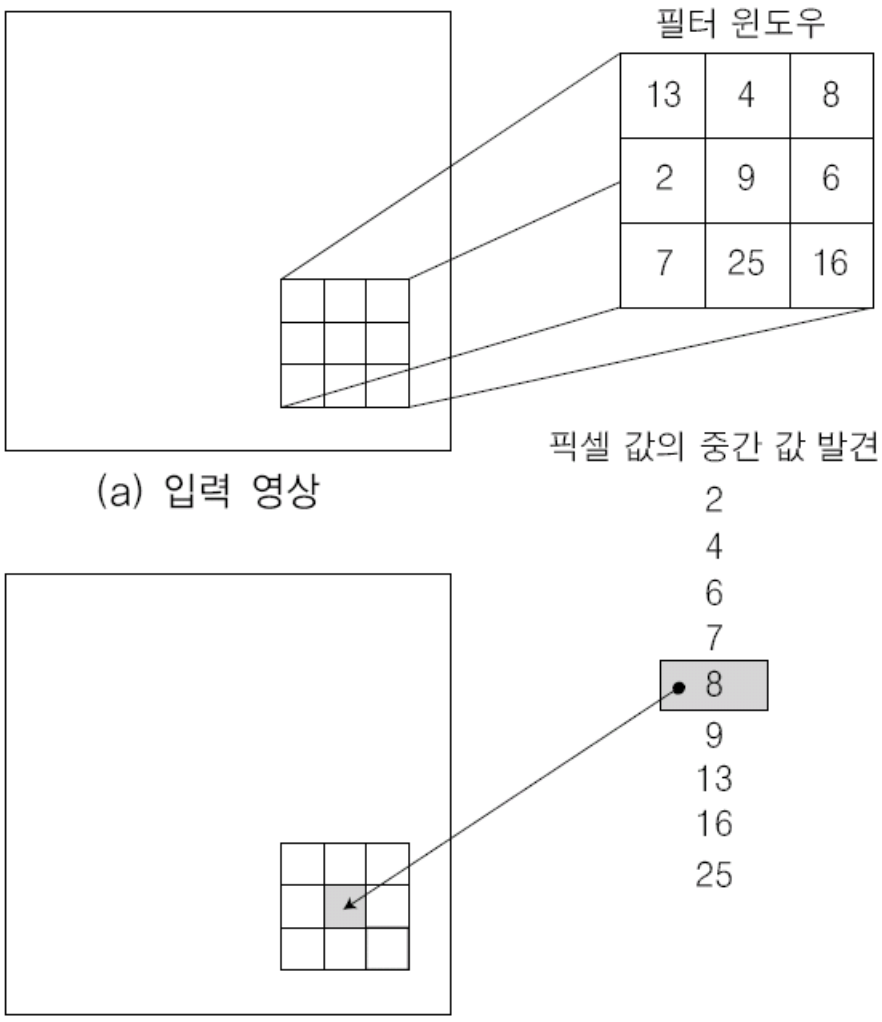


[그림 12-15] 다양한 중간 값 필터 마스크

중간 값 필터의 잡음제거 동작

- 중간 값 필터는 영상에 스파크처럼 급격한 색 변화가 있는 임펄스 잡음을 제거하는 데 사용
- 장점
  - 기존의 평균 필터를 이용한 선형 공간 필터링 방법에 비해 블러링 현상이 적고 객체의 경계를 잘 보존함
  - 즉, 평균 필터를 이용한 방법의 단점을 보완한 방법
- 단점
  - 중간 값을 구하려고 비교하는 과정에서 많은 시간이 소모됨.

중간 값 필터의 잡음제거 동작(계속)



[그림 12-16] 중간 값 필터 회선 마스크의 원리

[실습하기 12-4] 중간 값 필터링으로 잡음 제거하는 프로그램

① **ResourceView 창에서 [Menu]-[IDR_IMAGETYPE] 더블클릭 → 메뉴 추가**

ID	ID_MEDIAN_FILTER
Caption	중간 값 필터링

② **[MFC ClassWizard] 대화상자를 이용해 추가된 메뉴에서 중간 값 필터링을 실행하는 함수 추가**

Class Name	Function Type	Function Name
View Class	void	OnMedianFilter
Doc Class	void	OnMedianFilter

③ **Doc 클래스에 다음 프로그램 추가**

[실습하기 12-4] 중간 값 필터링으로 잡음 제거하는 프로그램

```
void CImageProcessingDoc::OnMedianFilter()
{
   int i, j, n, m, index = 0;
   double **tempInputImage, Mask[9];

   m_Re_height = m_height;
   m_Re_width = m_width;
   m_Re_size = m_Re_height * m_Re_width;

   m_OutputImage = new unsigned char [m_Re_size];

   tempInputImage = Image2DMem(m_height + 2, m_width + 2);

   for(i=0 ; i<m_height ; i++){
      for(j=0 ; j<m_width ; j++){
         tempInputImage[i+1][j+1]
            = (double)m_InputImage[i * m_width + j];
      }
   }
```

[실습하기 12-4] 중간 값 필터링으로 잡음 제거하는 프로그램

```
for(i=0 ; i<m_height ; i++){
      for(j=0 ; j<m_width ; j++){
         for(n=0 ; n<3 ; n++){
            for(m=0 ; m<3 ; m++){
               Mask[n*3 + m] = tempInputImage[i+n][j+m];
               // 3*3 크기 배열 값을 마스크 배열에 할당
            }
         }
         OnBubleSort(Mask, 9); // 마스크 값을 크기순으로 정렬
         m_OutputImage[index] = (unsigned char)Mask[4];
         // 중간 값 출력
         index++; // 출력 배열의 좌표
      }
   }
}
```

[실습하기 12-4] 중간 값 필터링으로 잡음 제거하는 프로그램

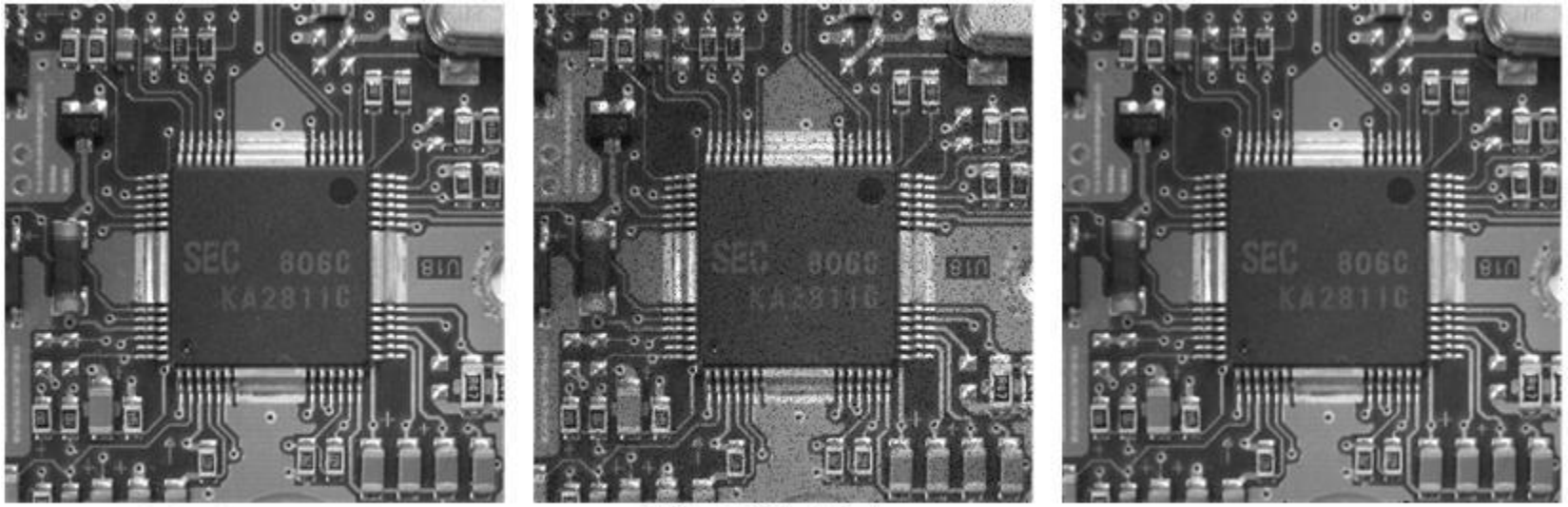
④ View 클래스에 다음 프로그램 추가

```
void CImageProcessingView::OnMedianFilter()
{
    CImageProcessingDoc* pDoc = GetDocument();
    ASSERT_VALID(pDoc);

    pDoc->OnMedianFilter();

    Invalidate(TRUE);
}
```

[실습하기 12-4] 중간 값 필터링으로 잡음 제거하는 프로그램

⑤ 프로그램 실행 결과 영상

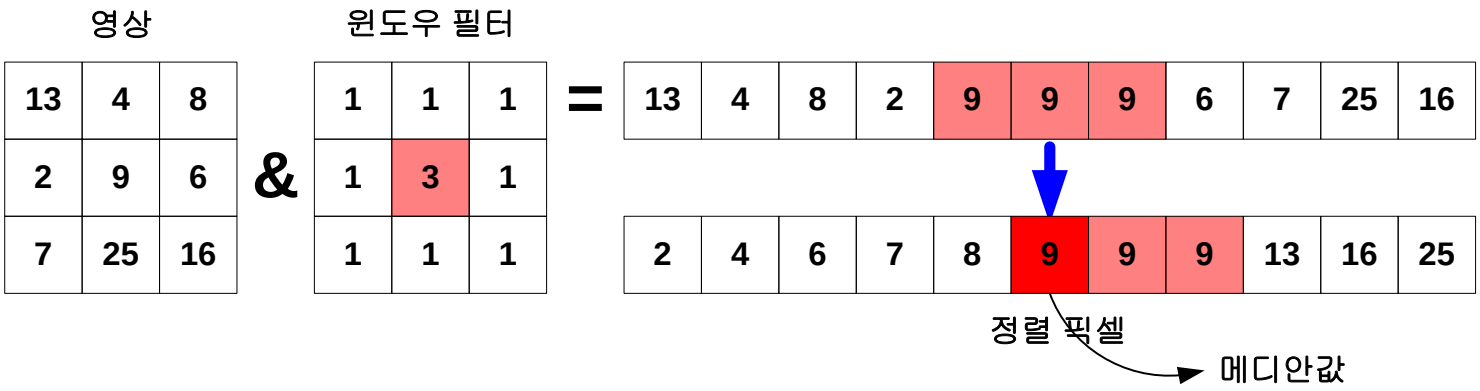
- (b)는 원본 영상에 임펄스 잡음을 첨가한 것
- (c)는 첨가된 임펄스 잡음을 중간 값 필터를 이용하여 제거한 결과 영상



(a) 원본 영상 (b) 잡음 첨가 (c) 중간 값 필터링

중간 값을 이용해 임펄스 잡음제거

- 가중 중간 값 필터
  - 중간 값 필터는 경계 부분을 잘 보존하는 편이지만, 좀더 세부적인 경계 영역까지도 보존할 수 있는 방법이 필요함. 이를 위해 표준 중간 값 필터를 확장한 가중 중간 값 필터(**Weighted Median Filter**)가 제안됨.
  - 이 방법은 가중치를 설정하여 영상 내의 세부 정보인 경계 영역을 보존하면서 동시에 잡음을 제거하는 특성이 있음.



가중 중간 값 필터의 원리

최소/최대 필터링으로 잡음제거

- 중심 화소를 이웃 화소의 중간 값으로 치환하는 대신 최소값이나 최대값으로 치환하는 방법을 최소/최대(Min/Max) 필터링이라고 함.
- 중간 값 필터링과 비슷한 방법
- 영상에 있는 극단적인 임펄스 값을 제거하는 데 사용되는 필터링 기법으로, 의료 영상에 주로 사용됨.
- 혼합된 임펄스 잡음을 제거하기는 어려움.

- 정렬된 값 중에서 최소값을 선택하는 최소값 필터링은 밝은 임펄스 값을 제거함.
  - 출력 영상의 전체 밝기가 감소
- 정렬된 값 중에서 최대값을 선택하는 최대값 필터링은 어두운 임펄스 값을 제거함
  - 출력 영상의 전체 밝기가 증가

[실습하기 12-5] 최대 필터링으로 잡음 제거하는 프로그램

① **ResourceView 창에서 [Menu]-[IDR_IMAGETYPE] 더블클릭 → 메뉴 추가**

ID	ID_MAX_FILTER
Caption	최대값 필터링

② **[MFC ClassWizard] 대화상자를 이용해 추가된 메뉴에서 최대값 필터링을 실행하는 함수 추가**

Class Name	Function Type	Function Name
View Class	void	OnMaxFilter
Doc Class	void	OnMaxFilter

③ **Doc 클래스에 다음 프로그램 추가**

[실습하기 12-5] 최대 필터링으로 잡음 제거하는 프로그램

```
void CImageProcessingDoc::OnMaxFilter()
{
   int i, j, n, m, index = 0;
   double **tempInputImage, Mask[9];

   m_Re_height = m_height;
   m_Re_width = m_width;
   m_Re_size = m_Re_height * m_Re_width;

   m_OutputImage = new unsigned char [m_Re_size];

   tempInputImage = Image2DMem(m_height + 2, m_width + 2);

   for(i=0 ; i<m_height ; i++){
      for(j=0 ; j<m_width ; j++){
         tempInputImage[i+1][j+1]
            = (double)m_InputImage[i * m_width + j];
      }
   }
```

[실습하기 12-5] 최대 필터링으로 잡음 제거하는 프로그램

```
for(i=0 ; i<m_height ; i++){
      for(j=0 ; j<m_width ; j++){
         for(n=0 ; n<3 ; n++){
            for(m=0 ; m<3 ; m++){
               Mask[n*3 + m] = tempInputImage[i+n][j+m];
               // 3*3 크기 배열 값을 마스크 배열에 할당
            }
         }
         OnBubleSort(Mask, 9); // 마스크 배열 값을 크기순으로 정렬
         m_OutputImage[index] = (unsigned char)Mask[8];
         // 최대값 출력
         index++; // 출력 배열의 좌표
      }
   }
}
```

[실습하기 12-5] 최대 필터링으로 잡음 제거하는 프로그램

④ View 클래스에 다음 프로그램 추가

```
void CImageProcessingView::OnMaxFilter()
{
    CImageProcessingDoc* pDoc = GetDocument();
    ASSERT_VALID(pDoc);

    pDoc->OnMaxFilter();

    Invalidate(TRUE);
}
```

[실습하기 12-5] 최대 필터링으로 잡음 제거하는 프로그램

⑤ 프로그램 실행 결과 영상

- 최대 필터링 영상 (b)는 영상의 밝기가 밝아졌고, 최소 필터링 영상 (c)는 영상의 밝기가 어두워졌음. 최대 필터와 최소 필터의 연속적인 수행은 혼합된 임펄스 잡음을 제거할 수 있음. 형태학 처리의 열림 및 닫힘연산과 비슷하게 개방형 필터(Opening Filter)와 폐쇄형 필터(Closing Filter)는 최대와 최소 필터를 순차적으로 수행. 폐쇄형 필터는 최대 필터를 수행한 뒤 최소 필터링이 적용, 개방형 필터는 최소 필터를 수행한 뒤 최대 필터를 적용.

(a) 원본 영상 (b) 최대값 필터링 (c) 최소값 필터링

최소/최대 필터 적용 결과

[실습하기 12-6] 최소 필터링으로 잡음 제거하는 프로그램

① **ResourceView 창에서 [Menu]-[IDR_IMAGETYPE] 더블클릭 → 메뉴 추가**

ID	ID_MIN_FILTER
Caption	최소값 필터링

② **[MFC ClassWizard] 대화상자를 이용해 추가된 메뉴에서 최소값 필터링을 실행하는 함수 추가**

Class Name	Function Type	Function Name
View Class	void	OnMinFilter
Doc Class	void	OnMinFilter

③ **Doc 클래스에 다음 프로그램 추가**

[실습하기 12-6] 최소 필터링으로 잡음 제거하는 프로그램

```
void CImageProcessingDoc::OnMinFilter()
{
   int i, j, n, m, index = 0;
   double **tempInputImage, Mask[9];

   m_Re_height = m_height;
   m_Re_width = m_width;
   m_Re_size = m_Re_height * m_Re_width;

   m_OutputImage = new unsigned char [m_Re_size];

   tempInputImage = Image2DMem(m_height + 2, m_width + 2);

   for(i=0 ; i<m_height ; i++){
      for(j=0 ; j<m_width ; j++){
         tempInputImage[i+1][j+1]
            = (double)m_InputImage[i * m_width + j];
      }
   }
```

[실습하기 12-6] 최소 필터링으로 잡음 제거하는 프로그램

```
for(i=0 ; i<m_height ; i++){
      for(j=0 ; j<m_width ; j++){
         for(n=0 ; n<3 ; n++){
            for(m=0 ; m<3 ; m++){
               Mask[n*3 + m] = tempInputImage[i+n][j+m];
               // 3*3 크기 배열 값을 마스크 배열에 할당
            }
         }
         OnBubleSort(Mask, 9); // 마스크 값을 크기순으로 정렬
         m_OutputImage[index] = (unsigned char)Mask[0];
         // 최소값 정렬
         index++; // 출력 배열의 좌표
      }
   }
}
```

[실습하기 12-6] 최소 필터링으로 잡음 제거하는 프로그램

④ View 클래스에 다음 프로그램 추가

```
void CImageProcessingView::OnMinFilter()
{
    CImageProcessingDoc* pDoc = GetDocument();
    ASSERT_VALID(pDoc);

    pDoc->OnMinFilter();

    Invalidate(TRUE);
}
```

요약

- 필터를 이용한 영상처리
  - 화소의 영역 처리에 해당
  - 컨벌루션으로 수행됨.
- 시스템
  - 일련의 입력 신호를 처리하여 또 다른 일련의 출력 신호를 만들어 내는 실체
  - 시스템의 성격에 따라 그 종류가 다양
- 선형 시불변 시스템(LTI)
  - 시스템을 설계하는 데 가장 적합
  - 선형이라는 특성과 시간에 따라 변하지 않는 시스템
- 필터 개념
  - 입력되는 신호의 일부 성분을 제거하거나 일부 특성을 변경하려고 설계된 하나의 시스템
- 필터 종류
  - 유한임펄스응답(FIR) 필터: 필터의 길이가 한정
  - 무한임펄스응답(IIR) 필터: 필터의 길이가 무한정
- 컨벌루션
  - 선형 시불변 시스템에 입력되는 신호가 어떤 신호를 출력하는지 결정해 줌.
- 필터링을 이용한 영상처리
  - 2차원의 컨벌루션을 수행함.
  - 2차원 컨벌루션은 회선

요약

- 영상처리에서 수행되는 회선 처리
  - 이웃 화소 각각에 회선 마스크의 가중치를 곱하여 더한 값으로 현재 화소를 변경하는 것
- 공간 주파수
  - 단위 공간에서 같은 화소 값이나 같은 색이 반복되는 횟수
  - 고주파: 변화가 빠르거나 색의 변화가 급격한 곳
  - 저주파: 밝기 변화가 늦거나 색의 변화가 적은 곳
- 공간 필터링
  - 영상에 있는 공간 주파수 대역을 제거하거나 강조하는 필터 처리
  - 사용되는 필터의 계수에 따라 특정 주파수를 제거하거나 강조하므로, 필터 마스크 또는 회선 마스크의 가중치 선택이 공간 필터의 행동을 결정함.
- 저주파 통과 필터(LPF)
  - 신호 성분 중 저주파 성분은 통과시키고 고주파 성분은 차단하는 필터
  - 잡음을 제거하거나 흐릿한 영상을 얻을 때 주로 사용
  - 고주파 성분을 제거하므로 고주파 차단 필터라고도 함
- 고주파 통과 필터(HPF)
  - 신호 성분 중 고주파 성분은 통과시키고 저주파 성분은 차단하는 필터
  - 저주파 성분을 차단하므로 저주파 차단 필터라고도 함.
- 고주파 강조 필터
  - 저주파 영역의 상쇄에 해당하는 부분에 일정량의 이득을 주어 낮은 공간 주파수에 해당하는 성분의 손실을 어느 정도 보상할 수 있음

요약

- 샤프닝 필터
  - 고주파 통과 필터에서 발생하는 낮은 공간 주파수 성분 손실 문제점을 보완해 주는 회선 마스크
  - 샤프닝 필터링된 영상은 원본 영상에 고주파 통과 필터링된 영상을 합한 것과 결과 비슷
- 저주파 통과 필터
  - 잡음에 해당하는 고주파 성분을 제거할 수 있음.
  - 선형 공간 필터링을 이용한 잡음제거 기법이라고도 함: 회선 마스크의 계수와 곱한 화소의 선형 합으로 연산을 수행하기 때문
- 평균 필터
  - 장점: 기준 화소 주변의 이웃 화소를 참조하여 평균 값으로 기준 화소 값을 변경하므로 영상 내의 급격한 변화를 나타내는 임펄스 잡음을 잘 제거
  - 단점: 전체에 블러링이 수행되어 원하지 않는 부분이 흐려짐.
- 비선형 공간 필터링
  - 필터 마스크의 상수 가중치를 곱한 화소의 선형적인 합으로 계산할 수 없는 방법
  - 이웃의 화소를 포함하는 비선형 연산을 바탕으로 한 공간 필터링
- 중간 값 필터(미디언 필터)
  - 이웃 화소의 값을 오름차순으로 정렬한 뒤 가운데에 있는 값을 출력 값으로 선택
  - 장점: 임펄스 잡음을 제거하는 데 사용되며, 블러링 현상이 적고 객체의 경계를 잘 보존함.
  - 단점: 중간 값을 구하려면 비교하는 과정에서 많은 시간이 소모됨.

요약

- 가중 중간 값 필터(Weighted Median Filter)
  - 가중치를 설정하여 영상 내의 세부 정보인 경계 영역을 보존하면서 동시에 잡음을 제거하는 특성이 있음.
- 최소/최대 필터링
  - 중간 값 필터링과 비슷한 방법인 중심 화소를 이웃 화소의 중간 값으로 치환하는 대신 최소값이나 최대값으로 치환하는 방법
  - 최소값 필터링: 정렬된 값 중에서 최소값을 선택. 밝은 임펄스 값을 제거→출력 영상의 전체 밝기가 감소
  - 최대값 필터링: 정렬된 값 중에서 최대값을 선택. 어두운 임펄스 값을 제거→출력 영상의 전체 밝기가 증가
- 폐쇄/개방형 필터링
  - 최대 필터와 최소 필터를 연속적으로 수행하면 혼합된 임펄스 잡음을 제거 가능
  - 폐쇄형 필터링: 최대 필터링→최소 필터링
  - 개방형 필터링: 최소 필터링→최대 필터링

Thank you