

# 영상처리 프로그래밍

---

Jongseok Lee([suk2080@kw.ac.kr](mailto:suk2080@kw.ac.kr))

Yong-Jo Ahn ([yjahn@digitalinsights.co.kr](mailto:yjahn@digitalinsights.co.kr))

2018-04-25

# Contents

---

- 5.1 실습 목적
- 5.2 실습 흐름도
- 5.3 실습 기초 이론
- 5.4 실습 따라 하기
- 5.5 실습 과제

# **[실습 5.1] 실습 목적**

---

## 5.1 실습 목적

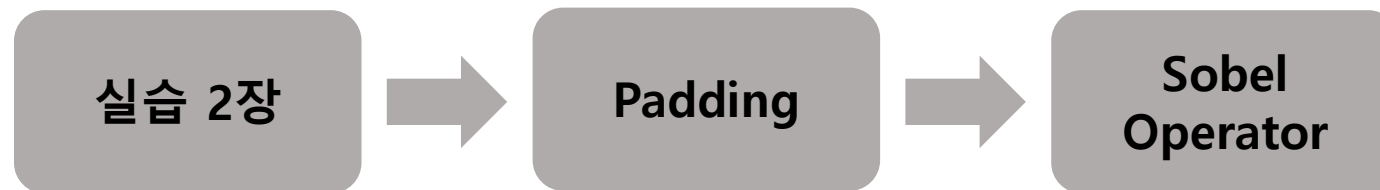
---

- 이번 장에서는 경계선을 검출하기 위해 고주파 통과 필터 중 하나인 Sobel operator를 이미지에 적용해본다.

# [실습 5.2] 실습 흐름도

## 5.2 실습 흐름도

---



# [실습 5.3] 실습 기초 이론

# 고주파 통과 필터

- 고주파 성분을 통과 시키는 반면 저주파 성분은 저지
- 경계선 추출
  - 경계점 밝기 값의 불연속을 찾는 방법.
  - 1차 미분치의 절대값이 큰 점을 불연속으로 정의. (밝기 값 변화, gradient)

$$\nabla f \equiv \text{grad}(f) \equiv \begin{bmatrix} g_x \\ g_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$$

$$M(x, y) \approx |g_x| + |g_y|$$



# Sobel operator

---

- 1차 미분치 값 계산 후 이를 임계치와 비교
- 수평과 수직 축으로 미분치 계산

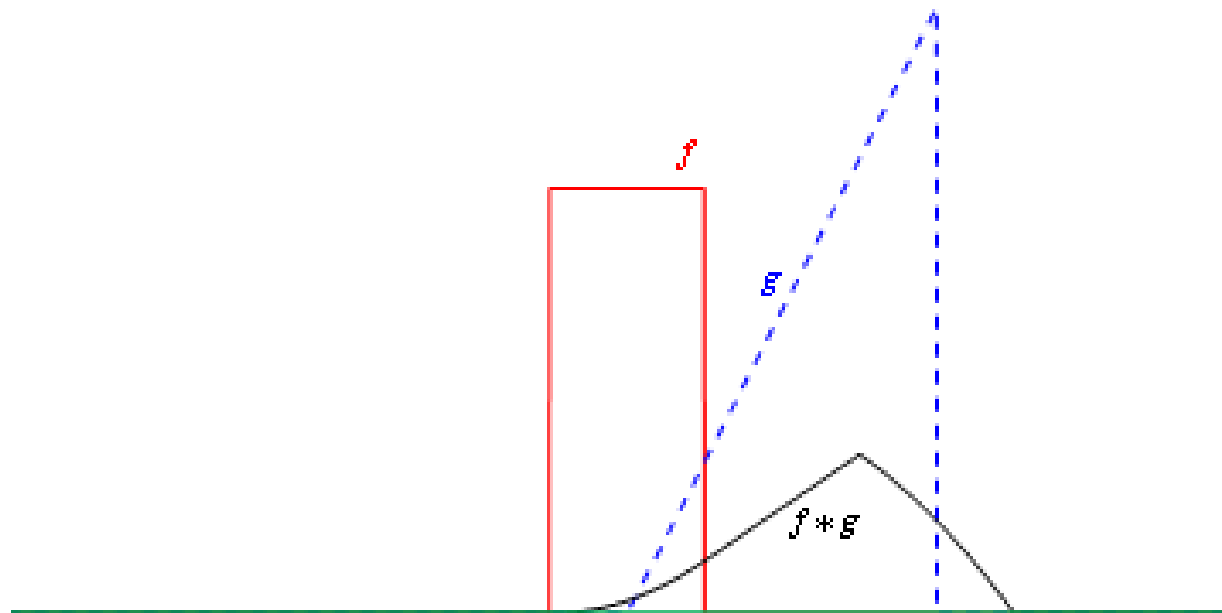
-1	-2	-1
0	0	0
1	2	1

1	0	-1
2	0	-2
1	0	-1

Sobel operator

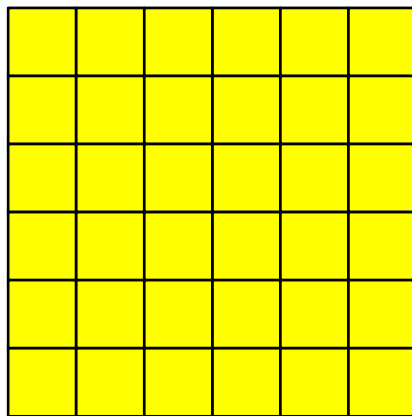
# 2D Convolution

- Convolution



# 2D Convolution

- 2-D discrete convolution

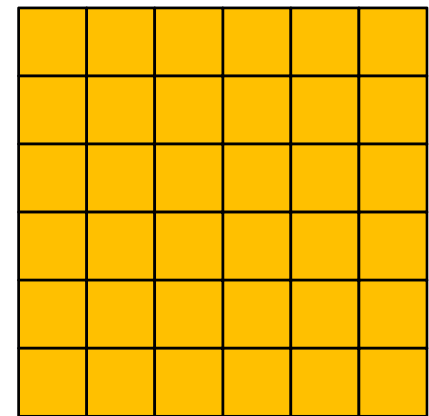


< Input image >



1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

< Mask >

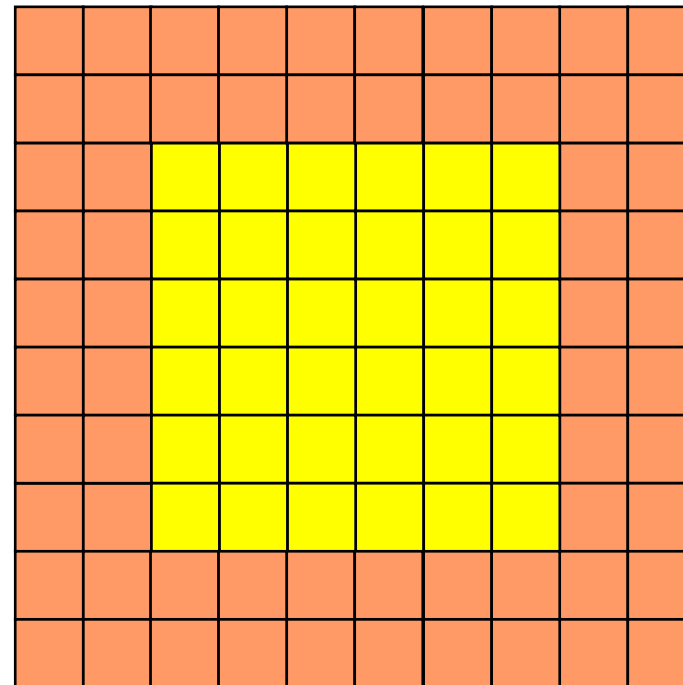
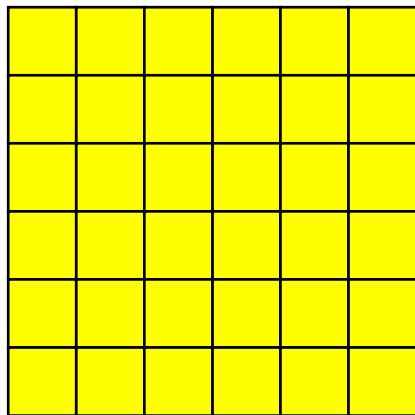


< Output image >

# 2D Convolution

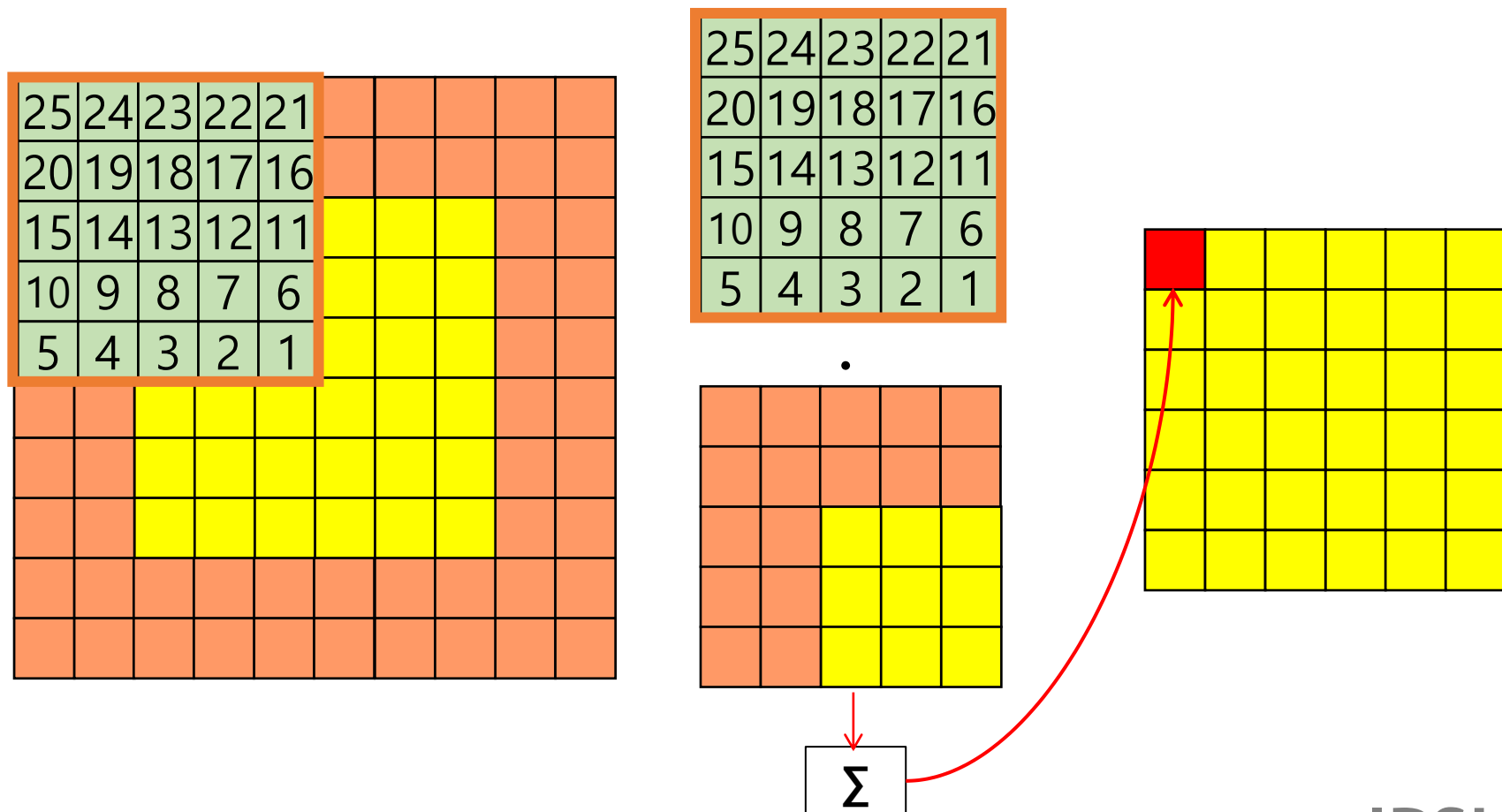
---

- 2-D discrete convolution
  - Image padding



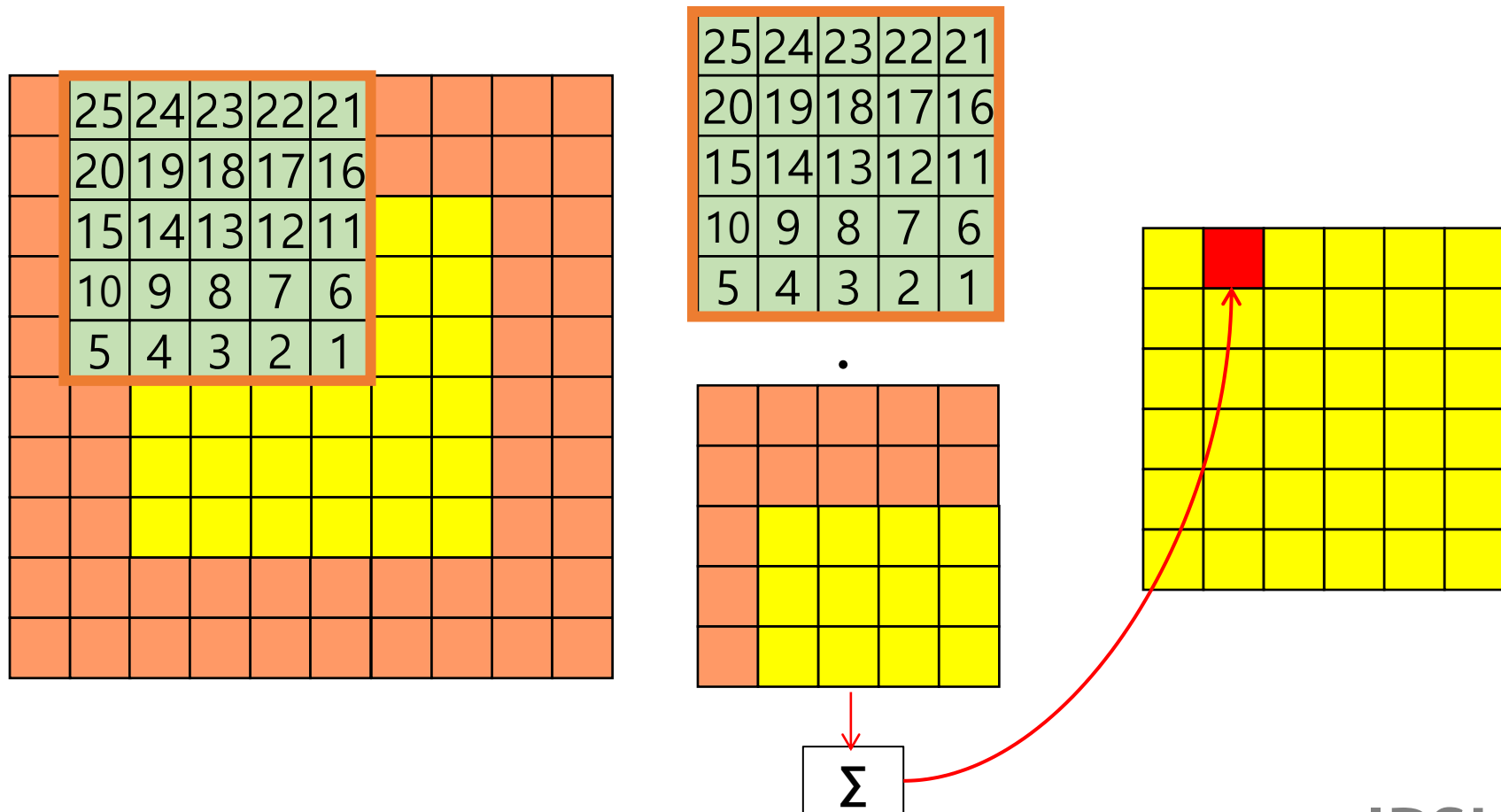
# Convolution

- 2-D discrete convolution



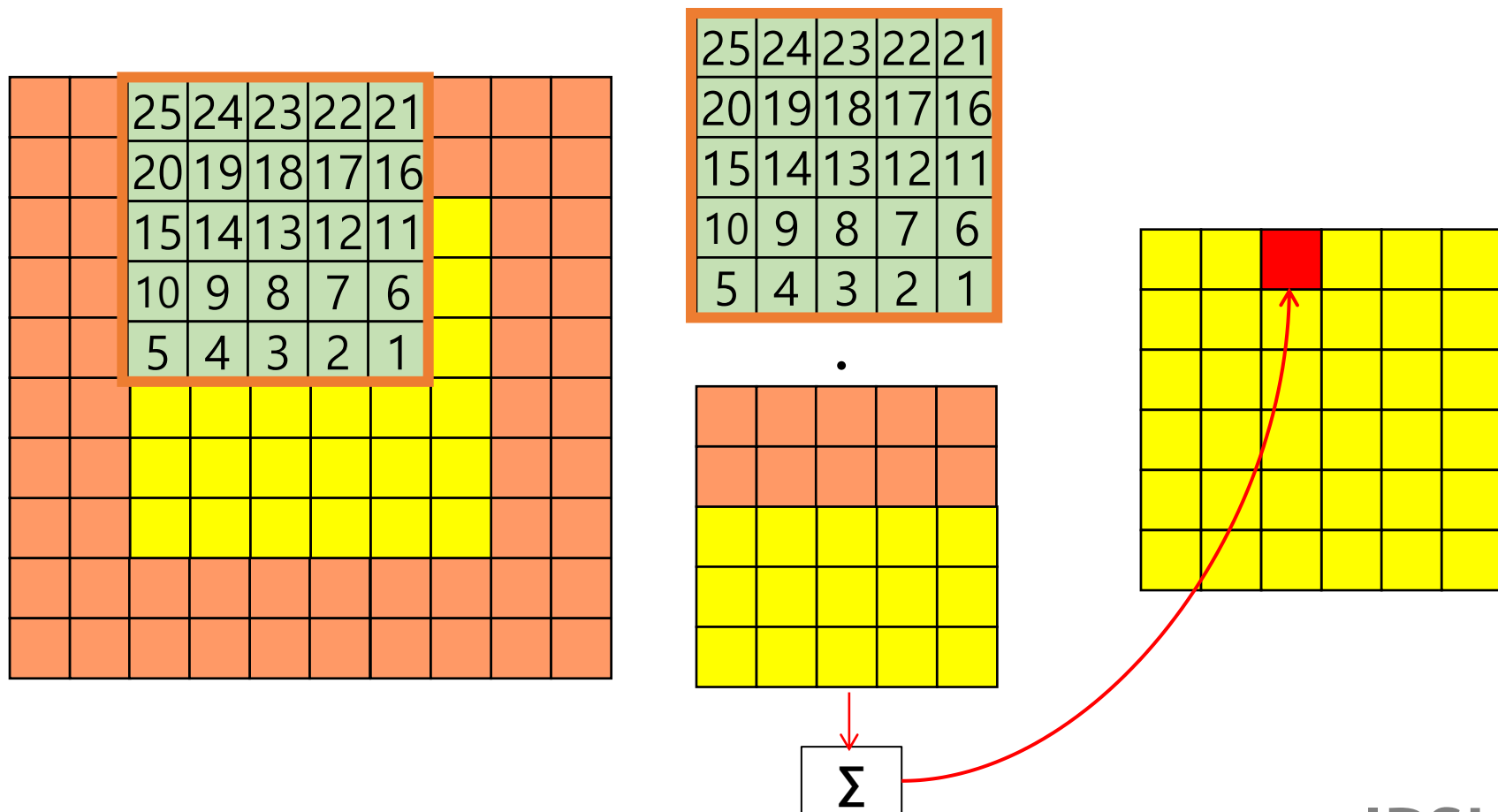
# Convolution

- 2-D discrete convolution



# Convolution

- 2-D discrete convolution

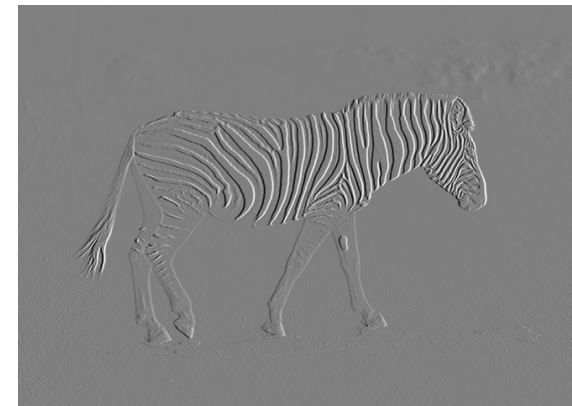


# Example of Sobel Operation

- Example (Sobel mask)

 $\frac{1}{8}$ 

-1	0	1
-2	0	2
-1	0	1

 $=$  $\frac{\partial f}{\partial x}$

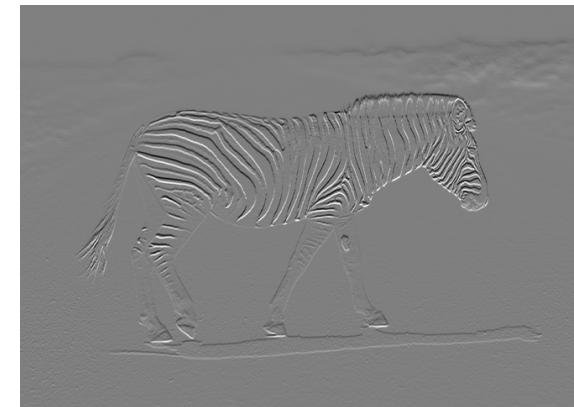


# Example of Sobel Operation

- Example (Sobel mask)

 $\frac{1}{8}$ 

-1	-2	-1
0	0	0
1	2	1

 $=$  $\frac{\partial f}{\partial y}$

# Example of Sobel Operation

---

- Example (Edge amplitude)



$$\|\nabla f\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$$

# Sobel operator

## ■ 의사코드

```
function sobel_operator is
  input :
    img_in[height][width]    //입력 이미지
    hori_filter[N][N]        // 수평 필터
    ver_filter[N][N]         // 수직 필터
  output :
    img_out[height][width]    // 출력 경계 영상

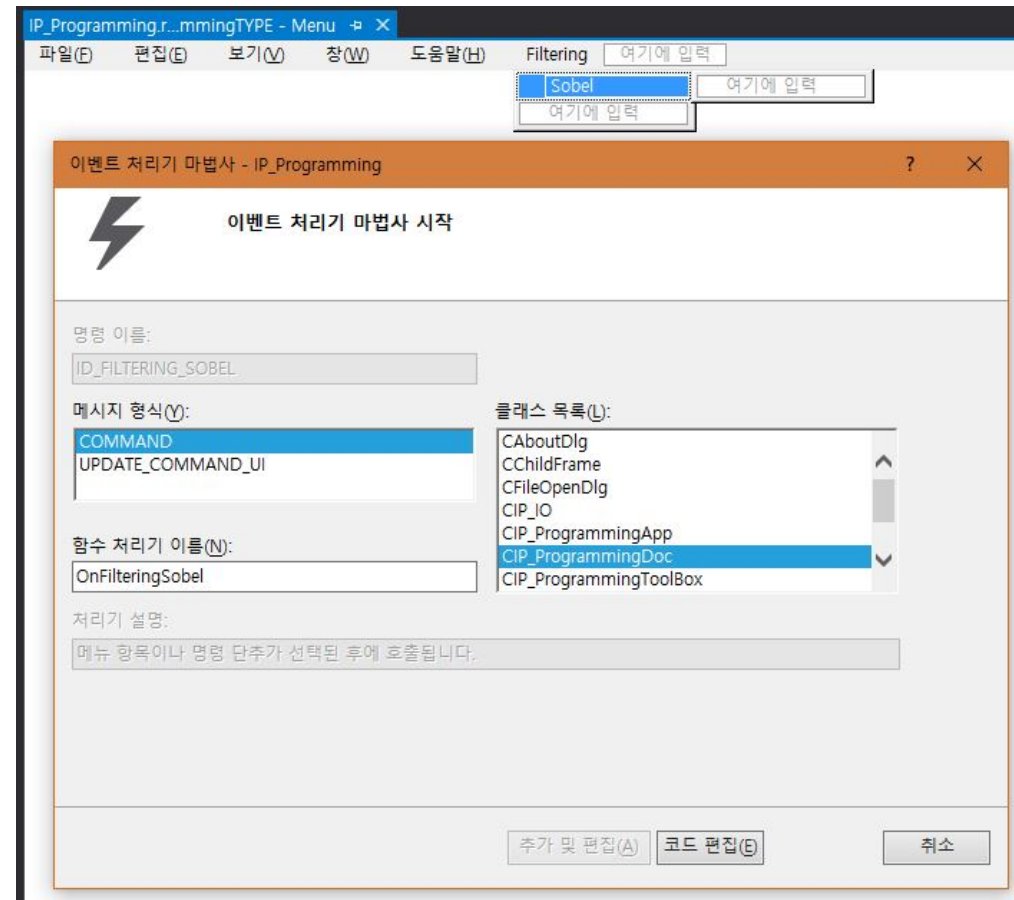
  for (i = N/2 to height - N/2) {
    for (j = N/2 to height - N/2) {
      ver_grad = 0;
      hori_grad = 0;
      for (y = -N/2 to N/2) {
        for (x = -N/2 to N/2) {
          hori_grad += hori_filter[y + N/2][x + N/2] * img_in[i+y][j+x];
          ver_grad += ver_filter[y + N/2][x + N/2] * img_in[i+y][j+x];
        }
      }

      if (abs(hori_grad) + abs(ver_grad) > threshold)
        img_out[i][j] = 255;
      else
        img_out[i][j] = 0;
    }
  }
end
```

# [실습 5.4] 실습 따라 하기

## 5.4.1 Sobel operator 이벤트 처리기

- Sobel Operator를 위한 메뉴와 해당 메뉴에 대한 이벤트 처리기 추가



## 5.4.2 Sobel operator 클래스

- CIP\_Sobel 클래스 추가
  - m\_pucSobelFilteringImgbuf : 검출된 경계 영상을 저장할 변수
  - MakePaddingImage : 경계 검출 전 영상을 패딩하는 함수
  - SobelFiltering : Sobel filtering을 수행하는 함수

```
IP_Sobel.h  X
IP_Programming
2 class CIP_Sobel
3 {
4 public:
5     UCHAR**      m_pucSobelFilteringImgbuf;
6
7 public:
8     CIP_Sobel();
9     ~CIP_Sobel();
10
11     UCHAR** memory_alloc2D(int height, int width);
12     int memory_free2D(UCHAR** ppMemAllocated);
13
14     UCHAR** MakePaddingImage(UCHAR** imgbuf, int Mask_Size, int height, int width);
15     void SobelFiltering(UCHAR**, int height, int width);
16 };
```

## 5.4.2 Sobel operator 클래스

- 추가한 변수와 함수에 대한 코드 작성
  - MakePaddingImage

```

IP_Sobel.cpp
IP_Programming
110 UCHAR** CIP_Sobel::MakePaddingImage(UCHAR** imgbuf, int Mask_Size, int height, int width)
111 {
112     int Pad_Size = Mask_Size >> 1;
113     int i, j;
114
115     UCHAR** Padding_Imgbuf = memory_alloc2D(height + (Pad_Size * 2), width + (Pad_Size * 2));
116     // TOP & BOTTOM
117     for (i = 0; i < Pad_Size; i++)
118     {
119         for (j = 0; j < width; j++)
120         {
121             // TOP
122             Padding_Imgbuf[i][j + Pad_Size] = imgbuf[0][j];
123             // BOTTOM
124             Padding_Imgbuf[height + Pad_Size + i][j + Pad_Size] = imgbuf[height - 1][j];
125         }
126     }
127     // LEFT & RIGHT
128     for (j = 0; j < Pad_Size; j++)
129     {
130         for (i = 0; i < height; i++)
131         {
132             // LEFT
133             Padding_Imgbuf[i + Pad_Size][j] = imgbuf[i][0];
134             // RIGHT
135             Padding_Imgbuf[i + Pad_Size][width + Pad_Size + j] = imgbuf[i][width - 3];
136         }
137     }
138
139     // CENTER
140     for (i = 0; i < height; i++)
141     {
142         for (j = 0; j < width; j++)
143         {
144             Padding_Imgbuf[i + Pad_Size][j + Pad_Size] = imgbuf[i][j];
145         }
146     }
147     // EDGE
148     for (i = 0; i < Pad_Size; i++)
149     {
150         for (j = 0; j < Pad_Size; j++)
151         {
152             // TOP-LEFT
153             Padding_Imgbuf[i][j] = imgbuf[0][0];
154             // BOTTOM-LEFT
155             Padding_Imgbuf[height + Pad_Size + i][j] = imgbuf[height - 1][0];
156             // TOP-RIGHT
157             Padding_Imgbuf[i][width + Pad_Size + j] = imgbuf[0][width - 3];
158             // BOTTOM-RIGHT
159             Padding_Imgbuf[height + Pad_Size + i][width + Pad_Size + j] = imgbuf[height - 1][width - 3];
160         }
161     }
162     return Padding_Imgbuf;
163 }

```

## 5.4.2 Sobel operator 클래스

- 추가한 변수와 함수에 대한 코드 작성
  - SobelFiltering

```

16 void CIP_Sobel::SobelFiltering(UCHAR** imgbuf, int height, int width)
17 {
18     int Mask_Size = 3;
19     int ImgPixel_x = 0;
20     int ImgPixel_y = 0;
21     int ImgPixel = 0;
22     int threshold = 150;
23
24     CHAR sobel_operator_y[3][3] =
25     {
26         { -1, -2, -1 },
27         { 0, 0, 0 },
28         { 1, 2, 1 },
29     };
30     CHAR sobel_operator_x[3][3] =
31     {
32         { 1, 0, -1 },
33         { 2, 0, -2 },
34         { 1, 0, -1 }
35     };
36
37     //원본 이미지 패딩
38     UCHAR** Padded_Imgbuf = MakePaddingImage( imgbuf,Mask_Size,height,width);
39     //필터링이 수행된 이미지를 저장
40     m_pucSobelFilteringImgbuf = memory_alloc2D(height, width);
41     //패딩된 이미지를 이용하여 필터링 수행
42     for (int i = 1; i < height; i++)
43     {
44         for (int j = 1; j < width; j++)
45         {
46             ImgPixel_x = 0;
47             ImgPixel_y = 0;
48             for (int m = 0; m < Mask_Size; m++)
49             {
50                 for (int n = 0; n < Mask_Size; n++)
51                 {
52                     //Mask 내의 모든 픽셀의 합 계산
53                     ImgPixel_x += Padded_Imgbuf[i + m - 1][j + n - 1] + sobel_operator_x[m][n];
54                     ImgPixel_y += Padded_Imgbuf[i + m - 1][j + n - 1] + sobel_operator_y[m][n];
55                 }
56             }
57             if (abs(ImgPixel_x) + abs(ImgPixel_y) > threshold)
58                 m_pucSobelFilteringImgbuf[i - 1][j - 1] = 255;
59             else
60                 m_pucSobelFilteringImgbuf[i - 1][j - 1] = 0;
61         }
62     }
63     //패딩된 영상 메모리 삭제
64     memory_free2D(Padded_Imgbuf);
65 }

```



## 5.4.2 Sobel operator 클래스

- 추가한 변수와 함수에 대한 코드 작성
  - 생성자, 소멸자

```
IP_Sobel.cpp  X
IP_Programming
4  CIP_Sobel::CIP_Sobel()
5      :m_pucSobelFilteringImgbuf(NULL)
6  {
7  }
8
9
10 CIP_Sobel::~CIP_Sobel()
11 {
12     if (m_pucSobelFilteringImgbuf)
13         memory_free2D(m_pucSobelFilteringImgbuf);
14 }
```

## 5.4.3 Sobel operator 출력

- 추가한 이벤트 처리기에 대한 코드 작성
  - Sobel operator 수행 후 경계가 검출된 이미지를 새 창에 띄우기 위해 onNewDocument 호출

```
IP_ProgrammingDoc.cpp ➦ X
IP_Programming
179 void CIP_ProgrammingDoc::OnFilteringSobel()
180 {
181     // TODO: 여기에 명령 처리기 코드를 추가합니다.
182     if (toolbox.io.m_Inputbuf == NULL)
183         return;
184     toolbox.sobel.SobelFiltering(toolbox.io.m_Inputbuf, toolbox.io.m_Height, toolbox.io.m_Width);
185     toolbox.io.m_Outputbuf = toolbox.sobel.m_pucSobelFilteringImgbuf;
186
187
188     CIP_ProgrammingApp *pApp = (CIP_ProgrammingApp*)AfxGetApp();
189     pApp->toolbox = &toolbox;
190     AfxGetMainWnd()->SendMessage(WM_COMMAND, ID_FILE_NEW);
191 }
```

## 5.4.3 Sobel operator 출력

- CIP\_ProgrammingDoc 클래스 OnNewDocument 함수 수정

```
IP_ProgrammingDoc.cpp ➔ X
IP_Programming
41 BOOL CIP_ProgrammingDoc::OnNewDocument()
42 {
43     if (!CDocument::OnNewDocument())
44         return FALSE;
45
46     CIP_ProgrammingApp *pApp = (CIP_ProgrammingApp*)AfxGetApp();
47     if (pApp->toolbox != NULL)
48     {
49         if (pApp->toolbox->io.m_Outputbuf)
50         {
51             toolbox.io.m_Height = pApp->toolbox->io.m_Height;
52             toolbox.io.m_Width = pApp->toolbox->io.m_Width;
53             toolbox.io.ID_MakeGrayImagetoBMP(pApp->toolbox->io.m_Outputbuf);
54
55             this->SetTitle("Output Image");
56         }
57         pApp->toolbox = NULL;
58     }
59     // TODO: 여기에 재초기화 코드를 추가합니다.
60     // SDI 문서는 이 문서를 다시 사용합니다.
61
62     return TRUE;
63 }
```

## 5.4.3 Sobel operator 출력

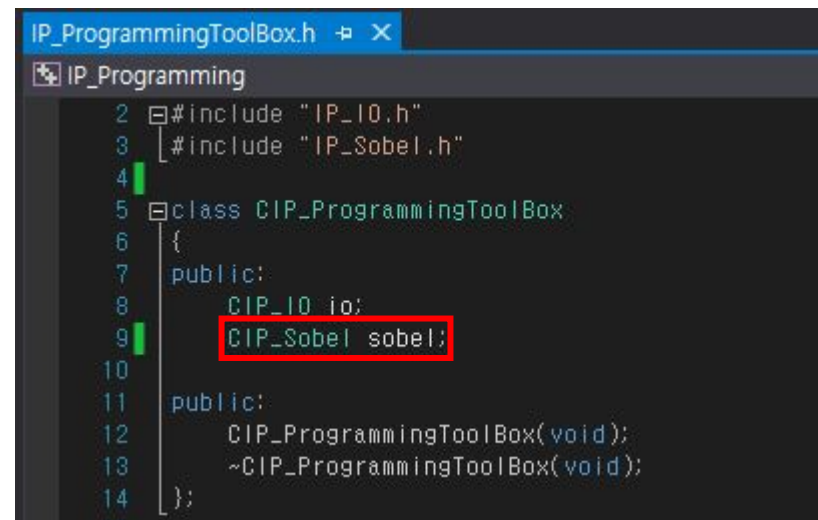
- CIP\_ProgrammingDoc 클래스 수정

```
IP_Programming.cpp  X
IP_Programming
34  CIP_ProgrammingApp::CIP_ProgrammingApp()
35      :toolbox(NULL)
36  {
```

```
IP_Programming.h  X
IP_Programming
13  // CIP_ProgrammingApp:
14  // 이 클래스의 구현에 대해서는 IP_Programming.cpp를 참조하십시오.
15  //
16  class CIP_ProgrammingToolBox;
17  class CIP_ProgrammingApp : public CWinApp
18  {
19  public:
20      CIP_ProgrammingApp();
21      ~CIP_ProgrammingApp();
22
23  public:
24      CIP_ProgrammingToolBox* toolbox;
25
26  // 재정의입니다.
27  public:
28      virtual BOOL InitInstance();
29      virtual int ExitInstance();
30
31  // 구현입니다.
32      afx_msg void OnAppAbout();
33      DECLARE_MESSAGE_MAP()
34  };
35
```

## 5.4.3 Sobel operator 출력

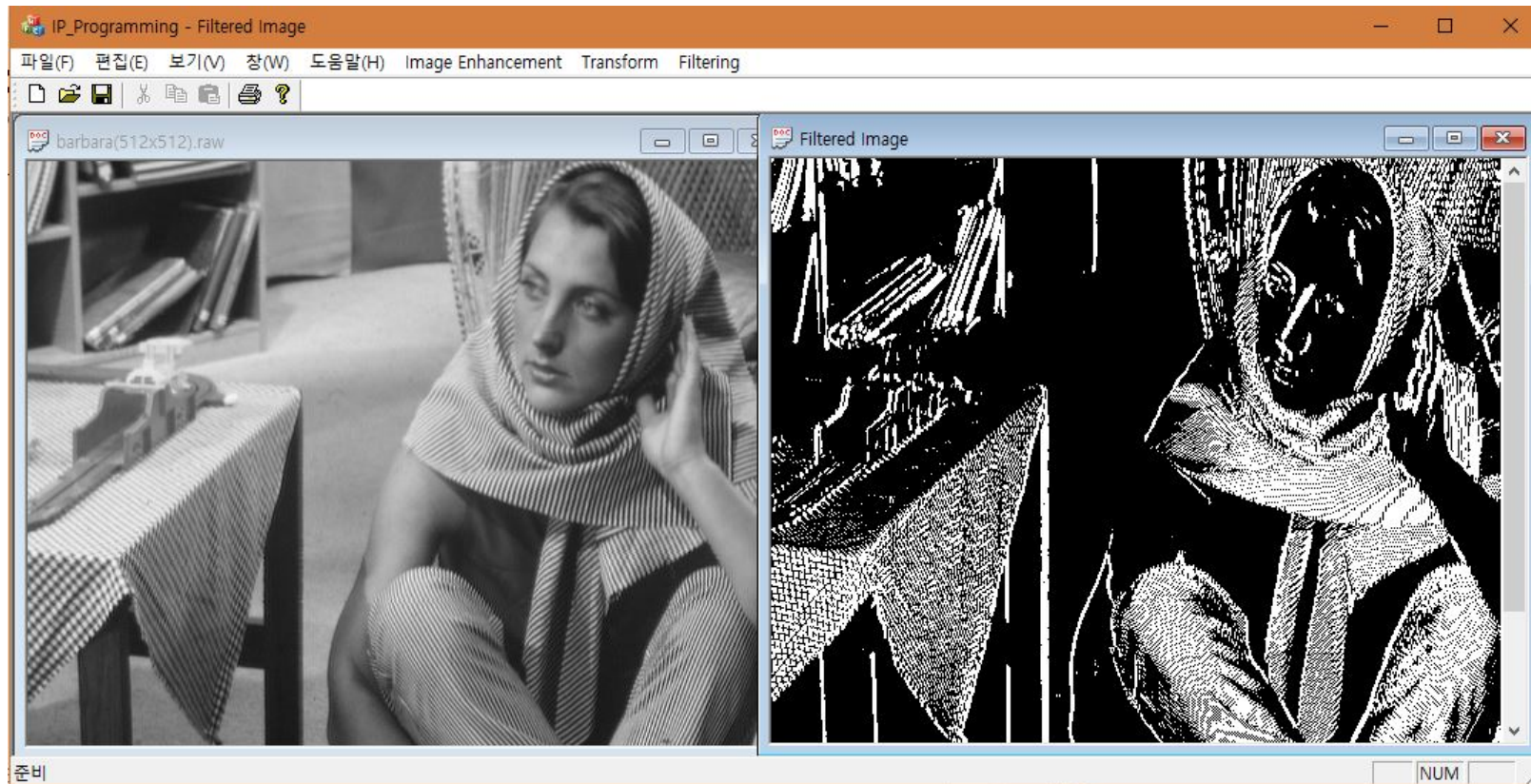
- CIP\_ProgrammingToolBox 클래스 수정



```
IP_ProgrammingToolBox.h
IP_Programming
2 #include "IP_IO.h"
3 #include "IP_Sobel.h"
4
5 class CIP_ProgrammingToolBox
6 {
7 public:
8     CIP_IO io;
9     CIP_Sobel sobel;
10
11 public:
12     CIP_ProgrammingToolBox(void);
13     ~CIP_ProgrammingToolBox(void);
14 };
15
```

## 5.4.4 Sobel operator 출력 결과

- 최종 출력 결과
  - 원본 이미지 (좌), Sobel Operator 적용 이미지 (우)

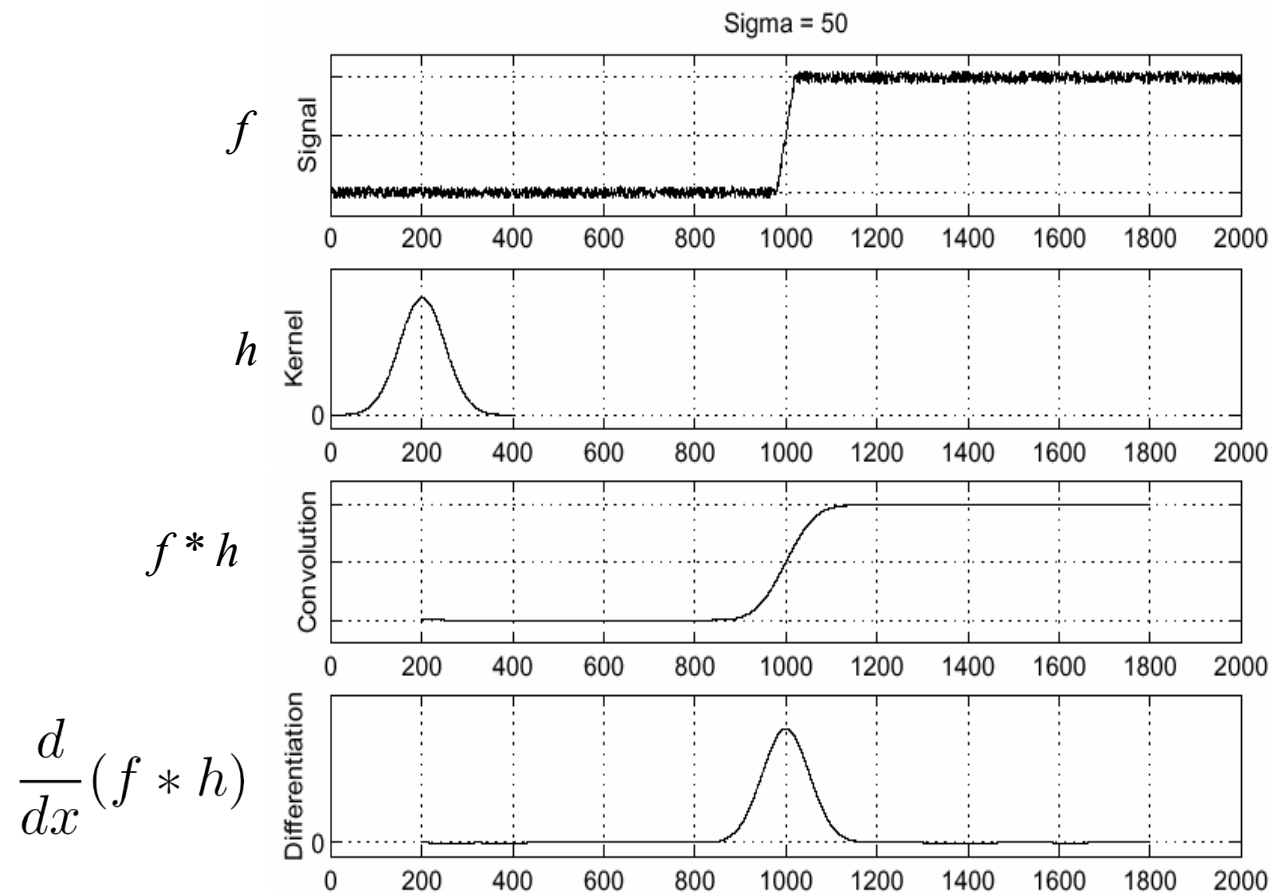


# **[실습 5.5] 실습 프로젝트**

---

## 5.5 실습 과제

- [실습 과제] LoG 필터
  - 입력 받은 영상에 대하여 Laplacian of Gaussian 필터링을 수행한다.





## 5.5 실습 과제

- Example (Laplacian mask)



1	1	1
1	-8	1
1	1	1



## 5.5 실습 과제

### ■ 2-Dimensional Gaussian function

$$G(x, y) = \frac{1}{\sqrt{2\pi\sigma_x^2}} \frac{1}{\sqrt{2\pi\sigma_y^2}} e^{-\left(\frac{(x-\mu_x)^2}{2\sigma_x^2} + \frac{(y-\mu_y)^2}{2\sigma_y^2}\right)}$$

where  $\left\{ \begin{array}{l} \mu_x : \text{expected value of X} \\ \mu_y : \text{expected value of Y} \\ \sigma_x^2 : \text{variance of X} \\ \sigma_y^2 : \text{variance of Y} \end{array} \right\}$

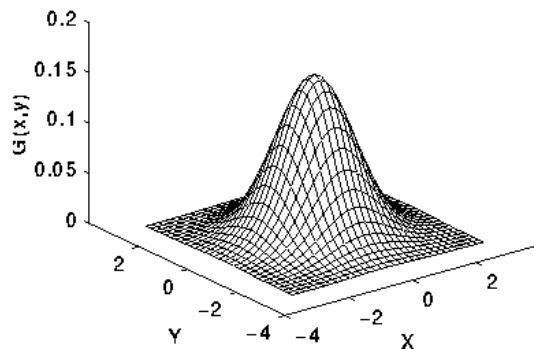
Circularly symmetric Gaussian

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \quad \text{where } \mu_x = \mu_y = 0, \quad \sigma_x^2 = \sigma_y^2 = \sigma^2$$

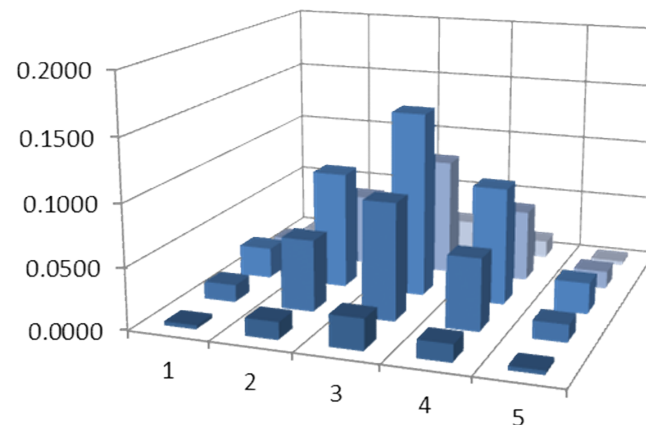
## 5.5 실습 과제

- 2-Dimensional Gaussian function
  - Circularly symmetric Gaussian

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \quad \text{where } \mu_x = \mu_y = 0, \quad \sigma_x^2 = \sigma_y^2 = \sigma^2$$



< 2-D Gaussian distribution with mean(0,0) and  $\sigma = 1$  >




0.0037	0.0147	0.0256	0.0147	0.0037
0.0147	0.0586	0.0952	0.0586	0.0147
0.0256	0.0952	0.1502	0.0952	0.0256
0.0147	0.0586	0.0952	0.0586	0.0147
0.0037	0.0147	0.0256	0.0147	0.0037

< Discrete approximation to 2-D Gaussian function with mean(0,0) and  $\sigma = 1$  >

## 5.5 실습 과제

- [실습 과제] LoG 필터
  - Laplacian of Gaussian (LOG)



$$* \left( \begin{array}{ccccc} 0.0037 & 0.0147 & 0.0256 & 0.0147 & 0.0037 \\ 0.0147 & 0.0586 & 0.0952 & 0.0586 & 0.0147 \\ 0.0256 & 0.0952 & 0.1502 & 0.0952 & 0.0256 \\ 0.0147 & 0.0586 & 0.0952 & 0.0586 & 0.0147 \\ 0.0037 & 0.0147 & 0.0256 & 0.0147 & 0.0037 \end{array} \right) * \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & -8 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array} \right) = ?$$

# END OF PRESENTATION

---

Q&A