미니 프로젝트(Version 1.0)

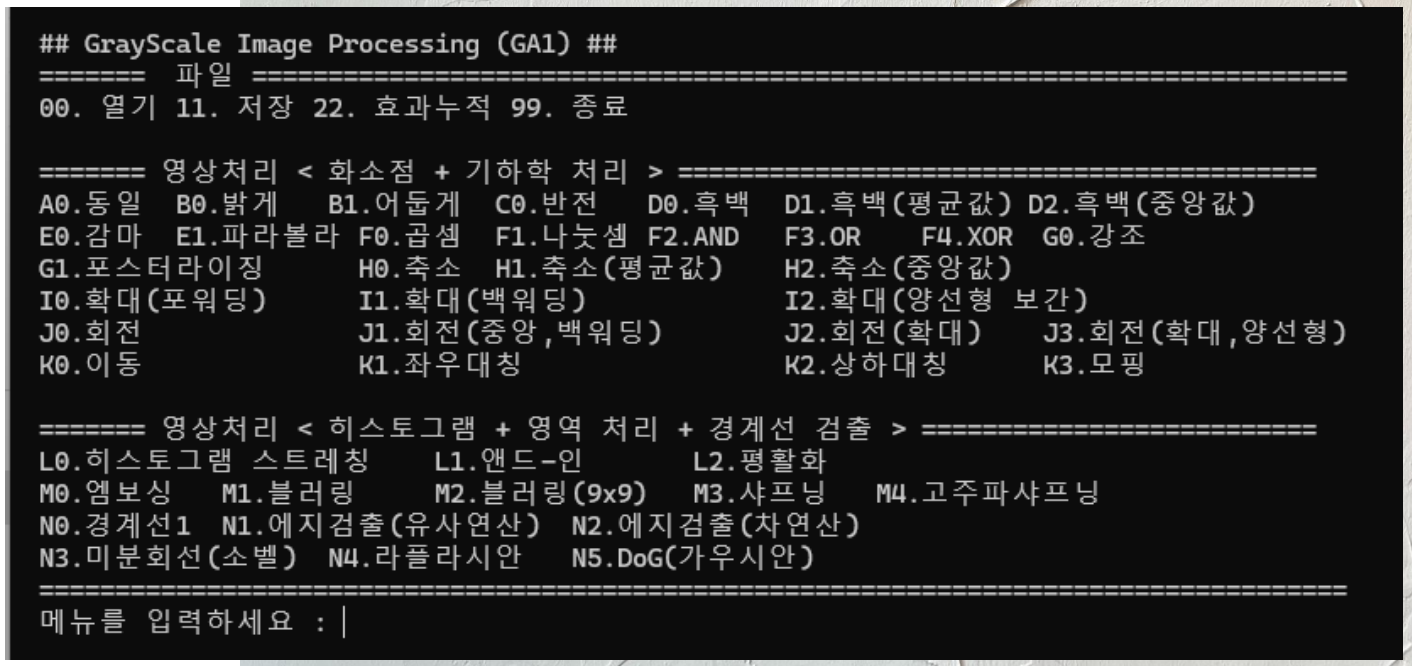
[Intel] 엣지 AI SW 아카데미
절차지향 프로그래밍

OpenCV 없이 C언어로 구현한 GrayScale Image Processing

이지원

프로젝트 개요

- 개발 목표: Open CV 없이 C언어로 영상처리
- 개발 기간: 2024.3.12~3.18
- 개발 환경: Visual Studio 2022, Windows 11
- 주요 기능:
  - 화소점 처리
  - 기하학 처리
  - 히스토그램 처리
  - 화소 영역 처리
  - 경계선 검출



프로그램 메뉴

```
## GrayScale Image Processing (GA1) ##
=======  파일 ================================================================
00. 열기 11. 저장 22. 효과누적 99. 종료

======= 영상처리 < 화소점 + 기하학 처리 > ==========================================
A0.동일  B0.밝게   B1.어둡게  C0.반전   D0.흑백  D1.흑백(평균값) D2.흑백(중앙값)
E0.감마  E1.파라볼라 F0.곱셈  F1.나눗셈 F2.AND  F3.OR    F4.XOR  G0.강조
G1.포스터라이징      H0.축소  H1.축소(평균값)   H2.축소(중앙값)
I0.확대(포워딩)      I1.확대(백워딩)          I2.확대(양선형 보간)
J0.회전              J1.회전(중앙,백워딩)      J2.회전(확대)    J3.회전(확대,양선형)
K0.이동              K1.좌우대칭              K2.상하대칭      K3.모핑

======= 영상처리 < 히스토그램 + 영역 처리 + 경계선 검출 > ========================
L0.히스토그램 스트레칭    L1.앤드-인       L2.평활화
M0.엠보싱   M1.블러링     M2.블러링(9x9)   M3.샤프닝   M4.고주파샤프닝
N0.경계선1  N1.에지검출(유사연산)  N2.에지검출(차연산)
N3.미분회선(소벨)  N4.라플라시안   N5.DoG(가우시안)
=============================================================================
메뉴를 입력하세요 :
```

- 파일: 열기, 저장, 효과 누적
- 화소점 처리: 15가지
- 기하학 처리: 13가지
- 히스토그램 처리: 3가지
- 화소 영역 처리: 4가지
- 경계선 검출: 6가지

1. 화소점 처리 (Point Processing)

- 화소 점의 원래 값이나 화소 점의 위치를 기준으로 화소 값을 변경하는 기술
- 다른 화소의 영향을 받지 않고 화소 점의 값만 변경
- 산술연산 - 밝게, 어둡게, 반전

원본 영상

밝게: 화소+50

어둡게: 화소-50

1. 화소점 처리

- 산술연산 (Cont.) - 상수 곱셈, 상수 나눗셈

곱셈연산

나눗셈연산

원본 영상

화소x1.5, 밝기값 증가

화소/1.5, 밝기값 감소

1. 화소점 처리

- 논리연산 - AND, OR, XOR, NOT

AND 연산

128로 AND 연산

OR 연산

128로 OR 연산

XOR 연산

입력이 다를 때만 1로
128로 XOR 연산

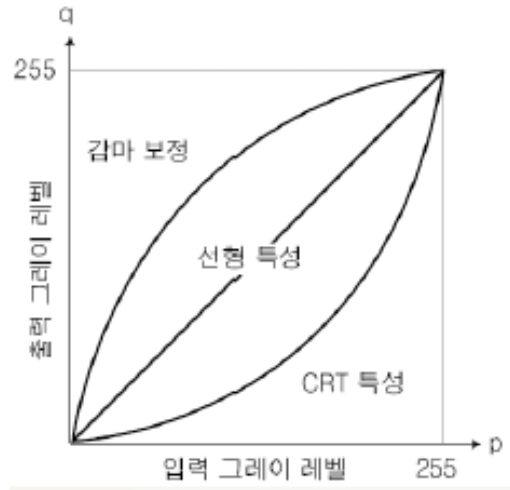
NOT 연산(반전)

검정색은 흰색으로,
흰색은 검정색으로

1. 화소점 처리

- 감마 보정(Gamma Correction)
  - 입력 값을 조정하여 출력을 만드는 과정
  - 함수의 감마 값에 따라 밝기 조정
  - 감마 값이 1보다 크면 영상이 어두워지고, 1보다 작으면 영상이 밝아짐

(d) 감마 보정 변환 함수 그래프




```
)255.0 * pow(temp / 255.0, gamma_val);
```

원본 영상

감마 값 0.7로 보정 영상

1. 화소점 처리

- 파라볼라: 영상에 입체감을 줌
  - CAP: 밝은 곳 입체형, CUP: 어두운 곳 입체형

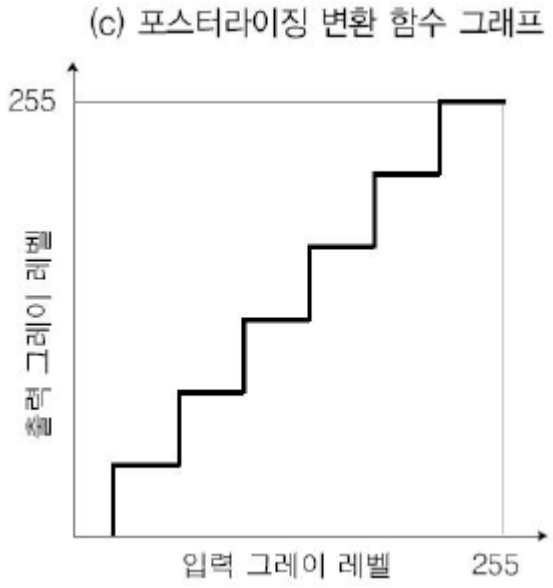
원본 영상

파라볼라 CUP 보정 영상

```
val = 255.0 * pow((temp / 128.0 - 1.0), 2);
```

1. 화소점 처리

- 포스터라이징
  - 명암 값의 범위를 경계 값으로 축소



원본 영상

명암값 256개가 6단계로 변환된 영상

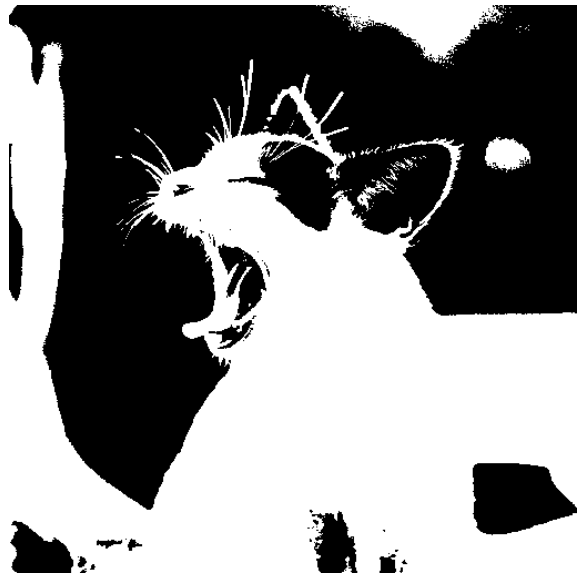
1. 화소점 처리

- 이진화 (Binarization)
  - 경계 값을 이용해 값이 두 개만 있는 영상으로 변환 $Output(q) = \begin{cases} 255 & Input(p) \geq T \\ 0 & Input(p) < T \end{cases}$

T = 128

T = 평균값

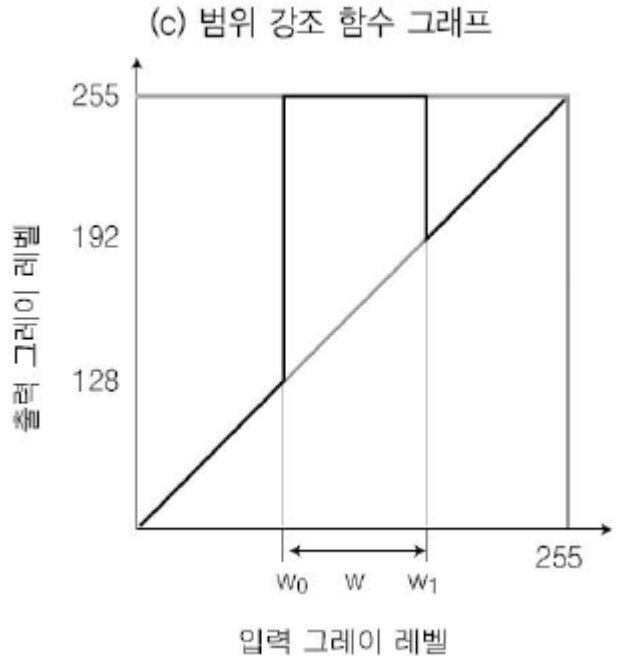
```
avg = sum / (outW * outH);
```



T = 중앙값
Quick Sort하여 중앙값 찾음

1. 화소점 처리

- 범위 강조 변환
  - 일정 범위의 화소만 강조하는 변환



원본 영상

강조 변환
W0=100, W1=140

2. 기하학 처리 (Geometric Processing)

- 화소의 공간적인 위치를 재배치하는 방법
- 선형 기하 변환: 직선 처리처럼 선형적으로 처리하는 방법
  - 이동, 회전, 스케일링 등 화소의 재배치
- 비선형 기하 변환 : 영상을 구부려서 곡선으로 처리하는 방법
  - 워핑과 모핑이 대표적

2. 기하학 처리

- 이동, 좌우대칭, 상하대칭

원본 영상

이동

좌우대칭
세로축을 중심으로 뒤집음

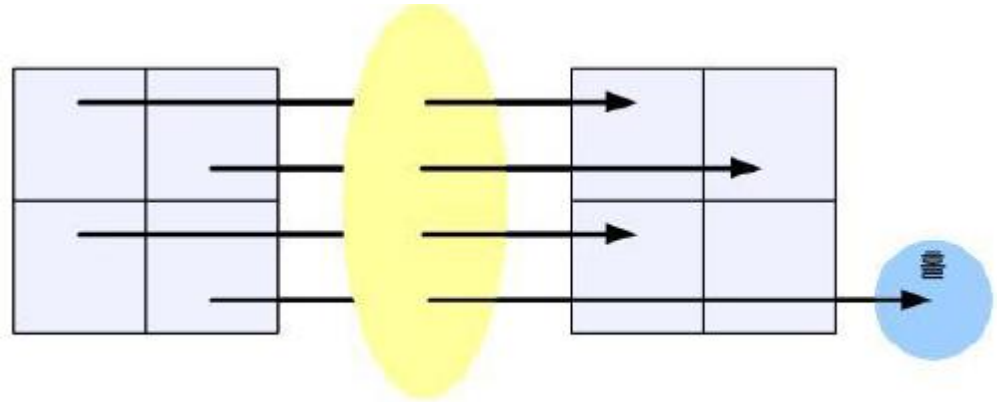
상하대칭
가로축을 중심으로 뒤집음

2. 기하학 처리

- 스케일링(Scaling)
  - 영상의 크기를 확대하거나 축소하는 변환
  - 원 영상의 해상도를 떨어뜨리는 특징이 있어 결과 영상의 품질이 떨어짐

2. 기하학 처리

- 확대(포워딩)
  - 전방향 사상
  - Hole 문제: 임의의 화소가 목적 영상의 화소에 사상되지 않음



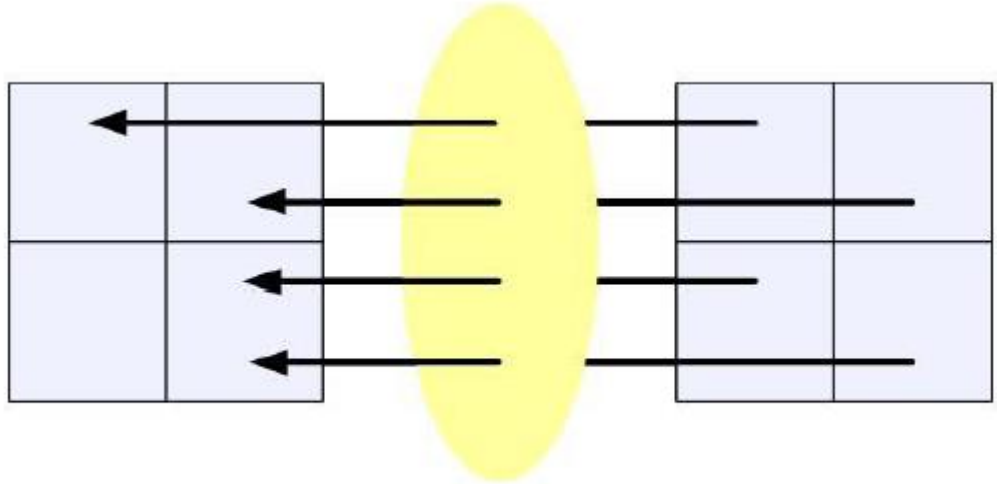
(b) 홀(hole) 문제

원본 영상

2배 확대, Hole 발생

2. 기하학 처리

- 확대(백워딩)
  - 역방향 사상: 홀 문제가 일어나지 않음



[그림 8-10] 역방향 사상의 동작

원본 영상

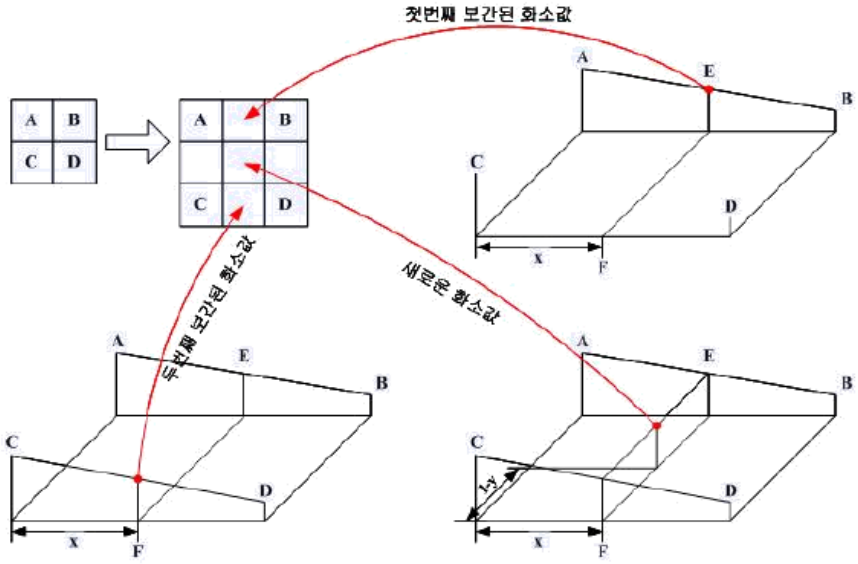
2배 확대, Hole 발생 X

2. 기하학 처리

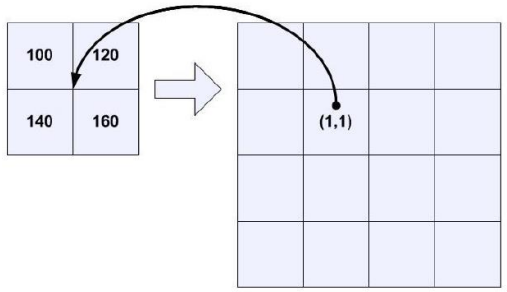
- 확대(양선형 보간법)
  - 보간법: 빈 화소에 값을 할당하여 좋은 품질의 영상을 만듬.
  - 선형 보간법: 화소 값 두개를 이용하여 새로운 화소값을 계산하는 방법
  - 양선형 보간법: 화소당 선형 보간을 세 번하여, 가장 가까운 화소 네 개에 가중치를 곱한 값을 합해서 얻음.

2. 기하학 처리

- 확대(양선형 보간법)



[그림 8-18] 보간을 세 번 수행하는 양선형 보간법



[그림 8-25] 양선형 보간법을 실제로 적용한 예

원본 영상

2배 확대, 양선형 보간

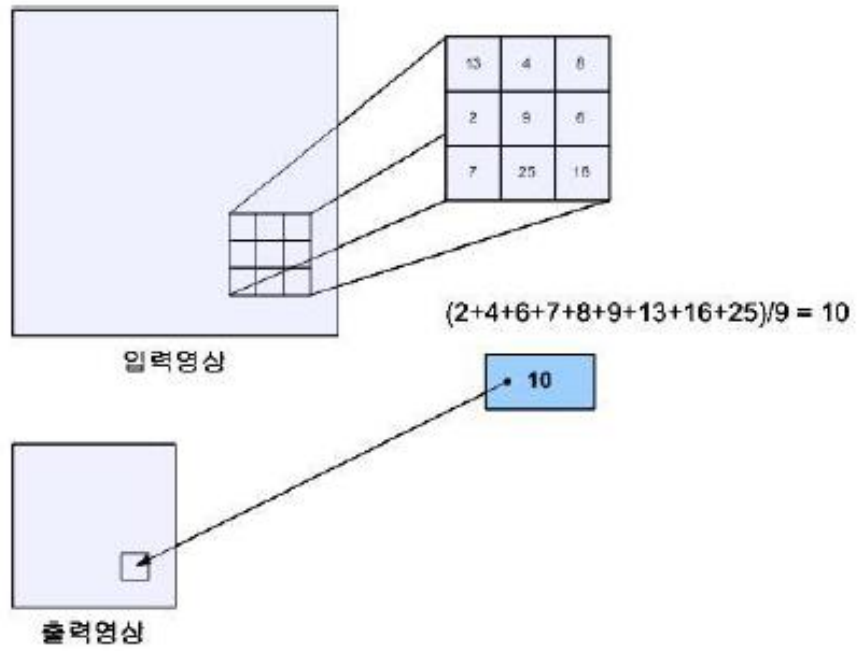
2. 기하학 처리

- 축소
  - 에일리어싱: 영상의 크기를 많이 축소하면, 세부 내용을 상실하게 되는 현상

2배 축소, 전방향사상

2. 기하학 처리

- 축소(평균값)

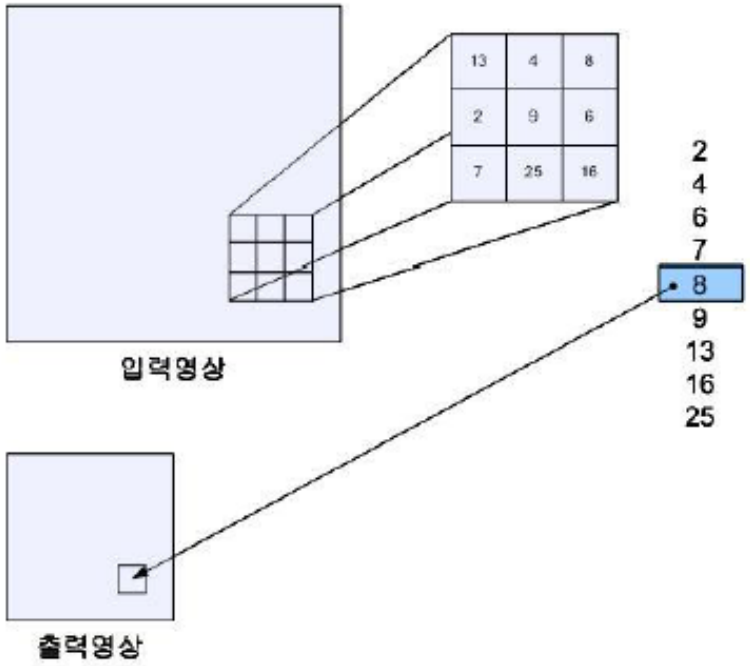


[그림 8-29] 평균 표현을 이용한 서브 샘플링의 동작

2배 축소

2. 기하학 처리

- 축소(중앙값)

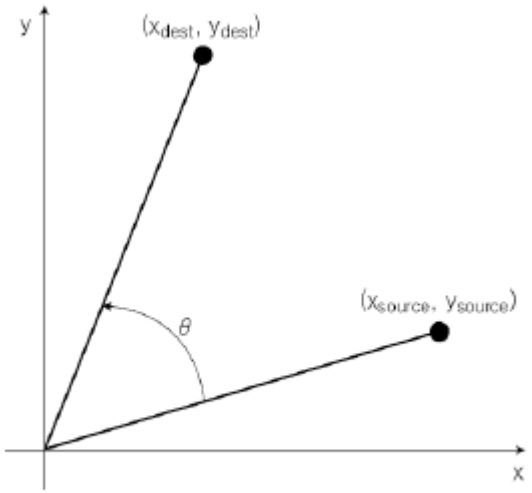


[그림 8-28] 미디언 표현을 이용한 서브 샘플링 동작 과정

2배 축소

2. 기하학 처리

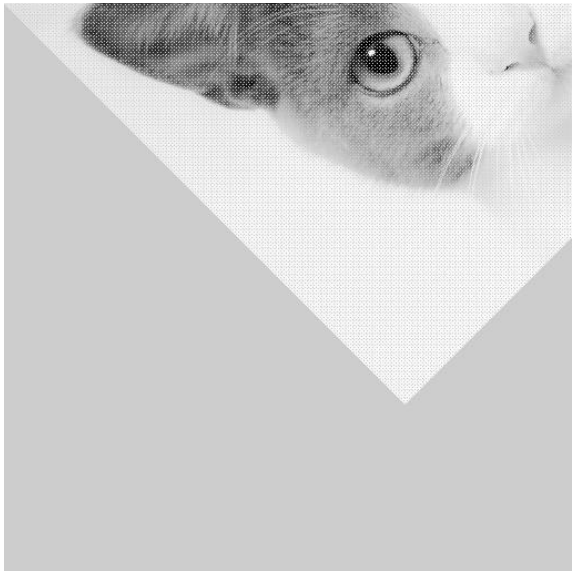
- 회전: 영상을 특정한 각도만큼 회전시키는 것 $\begin{bmatrix} x_{dest} \\ y_{dest} \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x_{source} \\ y_{source} \end{bmatrix}$



[그림 9-6] 회전 변환하여 좌표 변화

원본 영상

45도 회전



전방향사상 시 홀문제 발생
원점 기준 회전

2. 기하학 처리

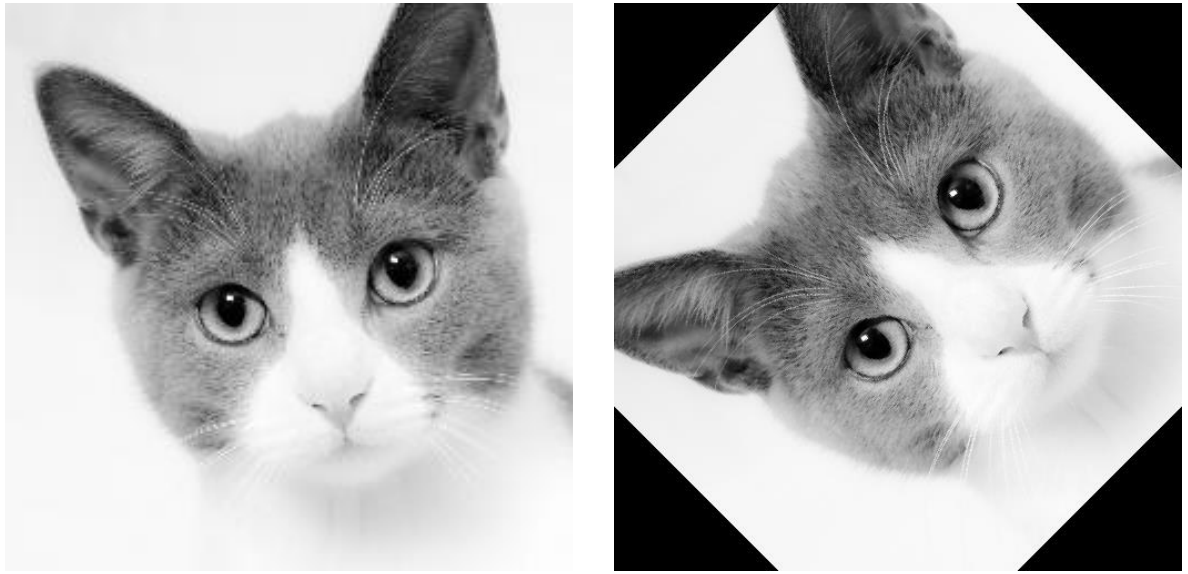
- 회전(중앙,백워딩) - 회전 결과 보이는 부분이 줄어드는 것을 방지
  - 역방향 사상: 홀 문제가 일어나지 않음

회전하려는 영상의 중심점이 **(Cx, Cy)**이고, 이 중심점을 기준으로 회전하는 전방향 사상 공식

$$\begin{bmatrix} x_{dest} \\ y_{dest} \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x_{source} - C_x \\ y_{source} - C_y \end{bmatrix} + \begin{bmatrix} C_x \\ C_y \end{bmatrix}$$

더 효율적으로 회전하기 위해 역방향 사상을 고려한 공식

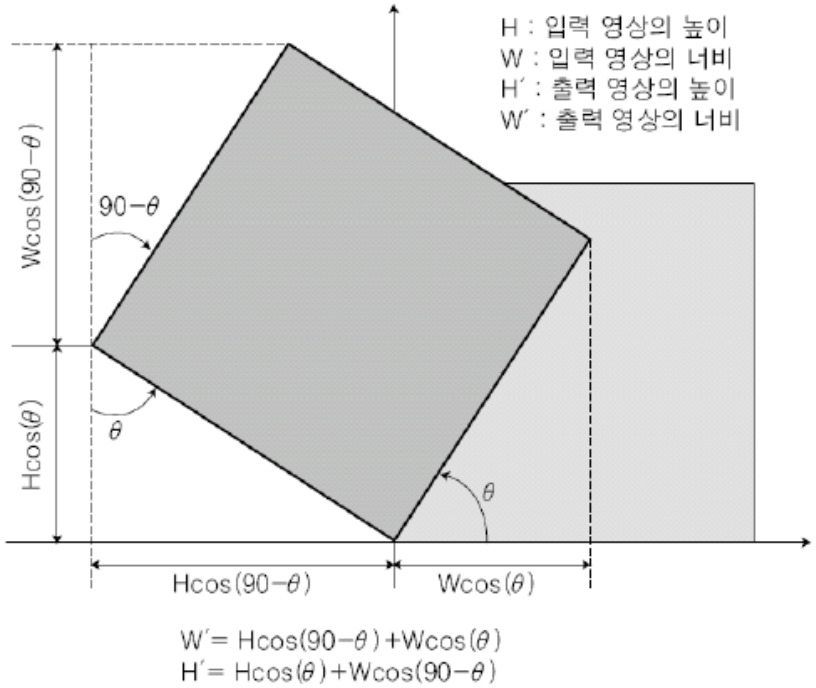
$$\begin{bmatrix} x_{source} \\ y_{source} \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x_{dest} - C_x \\ y_{dest} - C_y \end{bmatrix} + \begin{bmatrix} C_x \\ C_y \end{bmatrix}$$



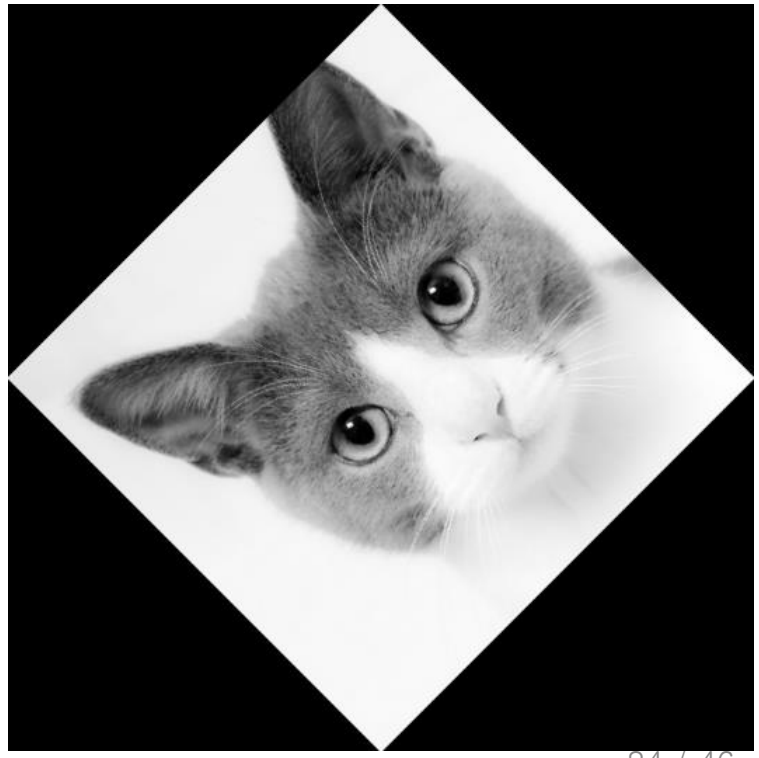
영상의 중심정을 기준으로 회전한 영상

2. 기하학 처리

- 회전(확대) : 출력 영상의 크기를 고려한 회전 변환
  - 출력 영상에서 잘려 나가는 부분이 없게 하려면 출력 영상 크기 계산



[그림 9-14] 회전 기하학적 변환의 출력 영상 크기



2. 기하학 처리

- 모핑: 한 영상을 서서히 다른 영상으로 변환하는 기술

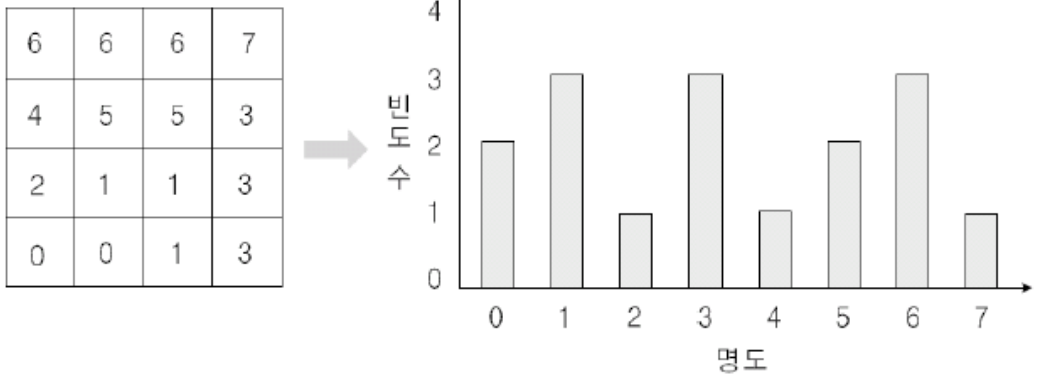
영상1

영상2

5:5로 모핑된 영상

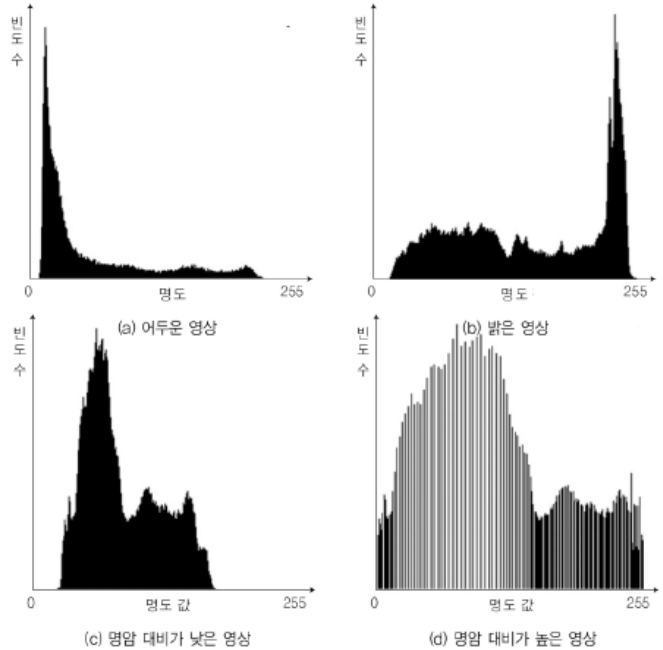
3. 히스토그램 처리

- 히스토그램
  - 관측한 데이터가 분포된 특징을 한눈에 볼 수 있도록 기둥 모양으로 나타낸 것
  - 가로축은 영상의 밝기, 세로축은 밝기 값에 대응하는 영상 내의 화소 수



(a) 입력 영상 (b) 히스토그램

[그림 5-1] 이상적인 영상의 히스토그램



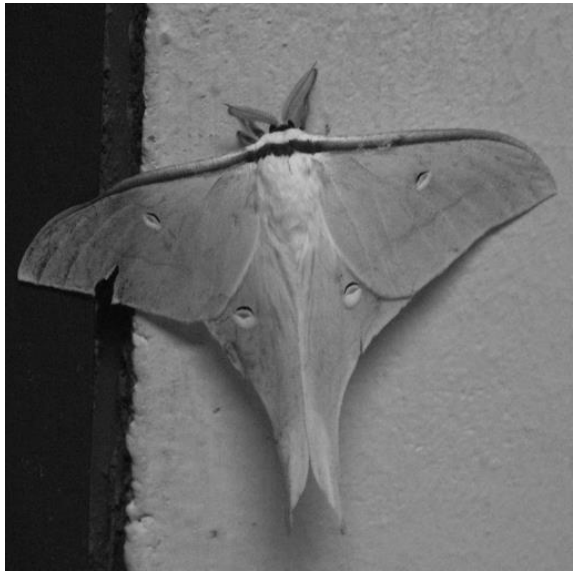
[그림 5-2] 영상의 특성에 따른 히스토그램

3. 히스토그램 처리

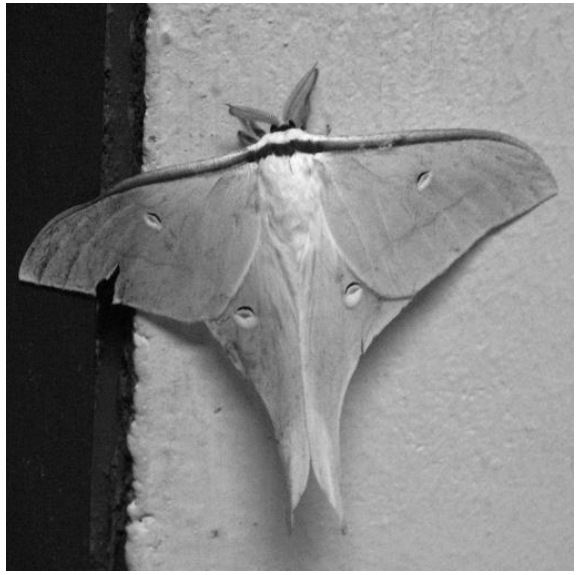
- 히스토그램 스트레칭 (Histogram Stretching)
  - 명암 대비를 향상시키는 연산

$$new\ pixel = \frac{old\ \ pixel - low}{high - low} \times 255$$

- old pixel은 원 영상 화소의 명도 값
- new pixel은 결과 영상 화소의 명도 값
- low는 히스토그램의 최저 명도 값
- high는 히스토그램의 최고 명도 값



원본 영상

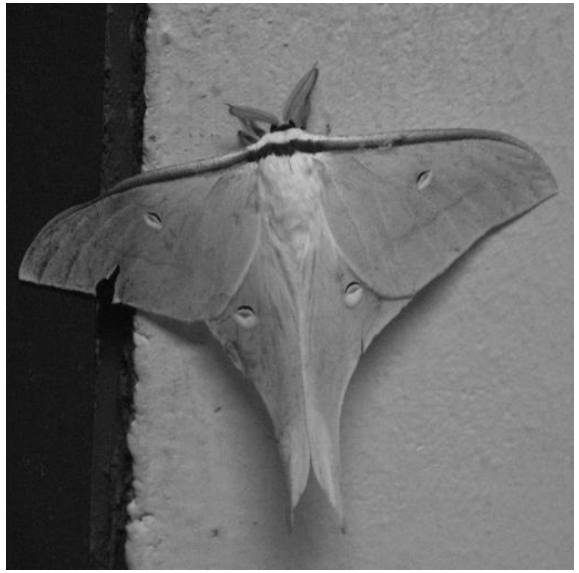


결과 영상

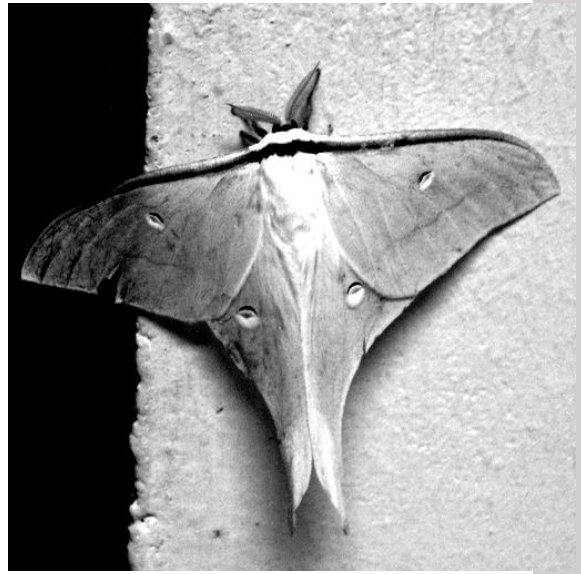
3. 히스토그램 처리

- 히스토그램 앤드-인
  - 두 개의 임계 값을 사용하여 히스토스램의 분포를 좀더 균일하게 만듬

$$new\ pixel = \begin{cases} 0 & old\ pixel \le low \\ \dfrac{old\ pixel - low}{high - low} \times 255 & low \le old\ pixel \le high \\ 255 & high \le old\ pixel \end{cases}$$



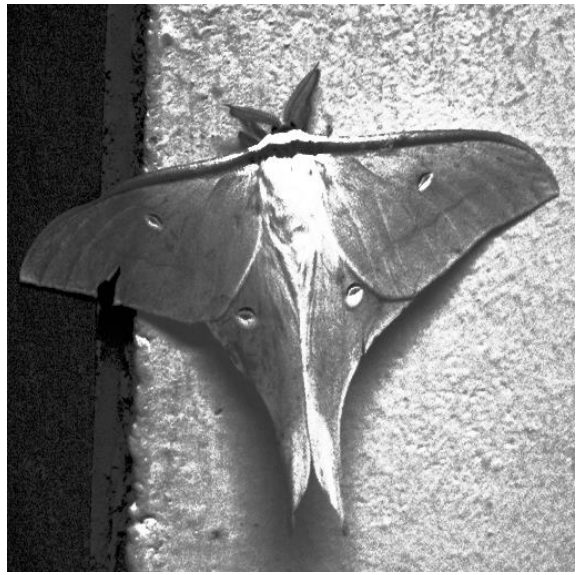
원본 영상



결과 영상

3. 히스토그램 처리

- 히스토그램 평활화 (Histogram Equalized)
  - 명암 분포가 빈약한 영상을 균일하게 만듬
  - 영상의 밝기 분포를 재분배하여 명함 대비를 최대
  - 1단계 : 히스토그램 빈도수 세기
  - 2단계 : 누적히스토그램 생성
  - 3단계 : 정규화된 히스토그램 생성
  - 4단계 : 입력 이미지를 정규화된 값으로 치환



결과 영상

4. 화소 영역 처리 (Area Processing)

- 화소의 원래 값과 이웃하는 화소의 값을 기반으로 화소값 변경하는 공간 영역 연산
- 원래 화소와 이웃한 각 화소의 가능치를 곱한 합을 출력 화소로 생성

$$Output\_pixel[x,y] = \sum_{m=(x-k)}^{x+k} \sum_{n=(y-k)}^{y+k} (I[m,n] \times M[m,n])$$

- **Output_pixel[x, y]**: 회선 처리로 출력한 화소
- **I[m, n]**: 입력 영상의 화소
- **M[m, n]**: 입력 영상의 화소에 대응하는 가중치

I_1	I_2	I_3
I_4	I_5	I_6
I_7	I_8	I_9

(a) 입력 영상

M_1	M_2	M_3
M_4	M_5	M_6
M_7	M_8	M_9

(b) 회선 마스크

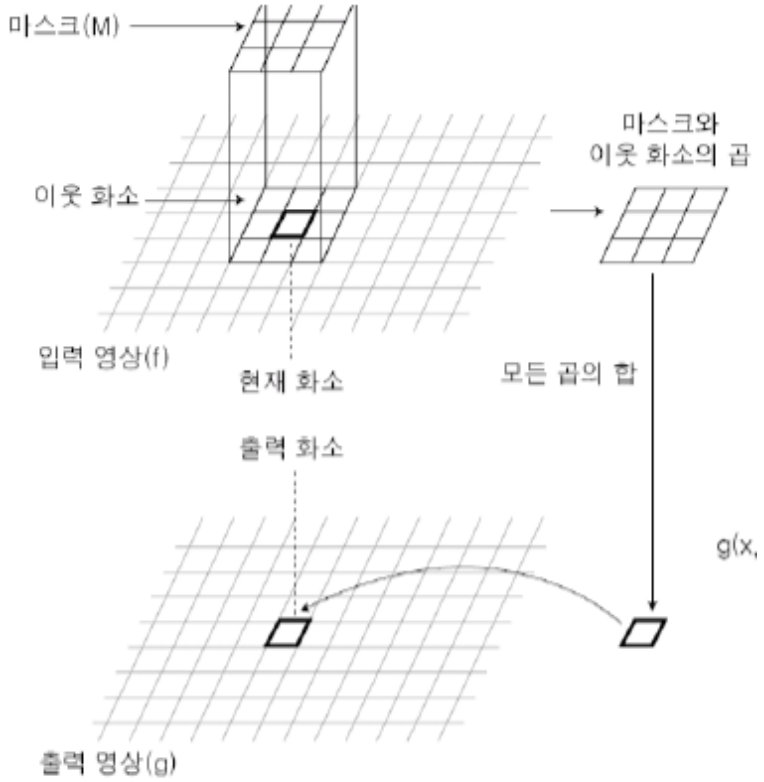
출력 픽셀 값 :

$$I_1 \times M_1 + I_2 \times M_2 + I_3 \times M_3 + I_4 \times M_4 + I_5 \times M_5 + I_6 \times M_6 + I_7 \times M_7 + I_8 \times M_8 + I_9 \times M_9$$

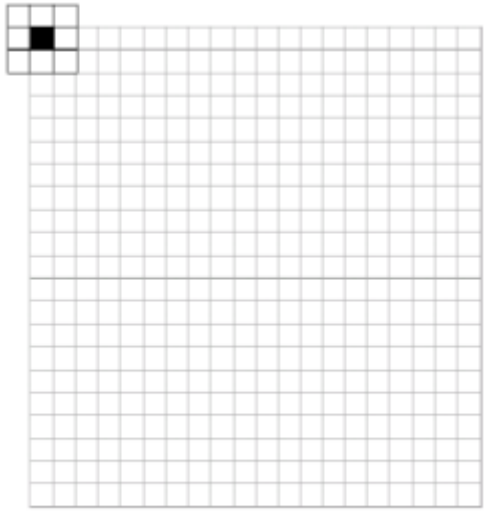
회선 기법으로 출력 화소 생성

4. 화소 영역 처리

- 가중치를 포함한 회선 마스크가 이동하면서 수행



[그림 6-7] 디지털 영상에서 회선을 처리하는 과정

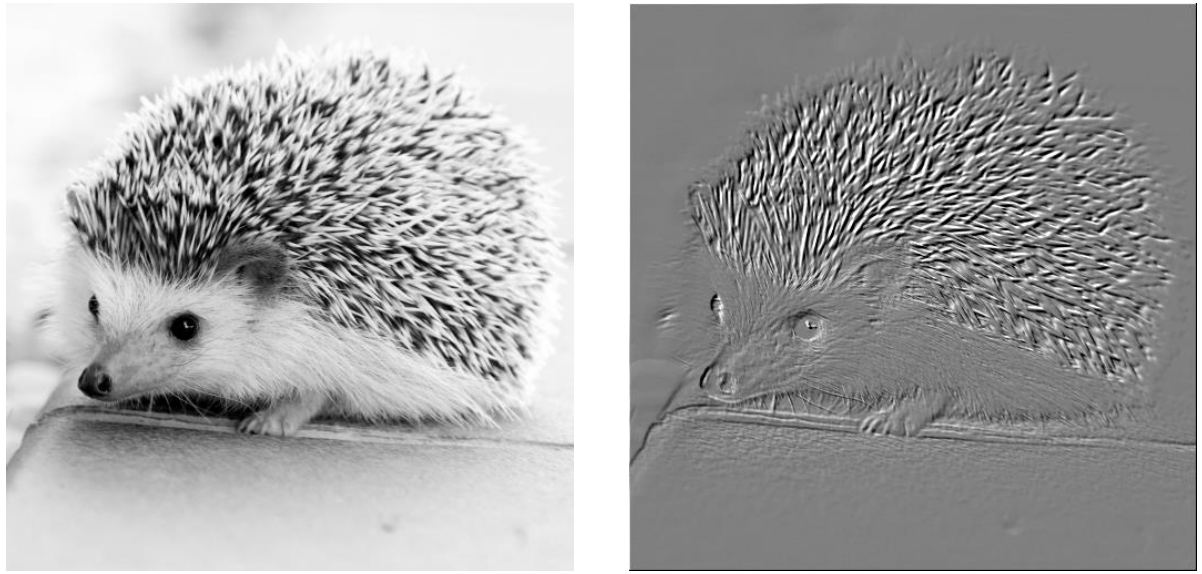


[그림 6-8] 회선 수행이 시작되는 위치

4. 화소 영역 처리

- 엠보싱 : 적절하게 구분된 경계선으로 영상이 볼록한 느낌을 가짐

```
double mask[3][3] = { {-1.0, 0.0, 0.0}, // 엠보싱 마스크
                      {0.0,  0.0, 0.0},
                      {0.0,  0.0, 1.0} };
```



결과 영상: 음각과 양각된 것처럼 느껴짐

4. 화소 영역 처리

- 블러링
  - 영상의 세밀한 부분을 제거하여 영상을 흐리거나 부드럽게 하는 기술
  - 영상의 세밀한 부분은 고주파 성분인데, 고주파 성분을 제거해 줌.
  - 블러링 회선 마스크는 모든 계수가 양수로 전체 합은 1

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

1/25	1/25	1/25	1/25	1/25
1/25	1/25	1/25	1/25	1/25
1/25	1/25	1/25	1/25	1/25
1/25	1/25	1/25	1/25	1/25
1/25	1/25	1/25	1/25	1/25

블러링 마스크의 회선 계수

3x3 마스크 적용

9x9 마스크 적용

4. 화소 영역 처리

- 샤프닝, 고주파 샤프닝
  - 영상의 상세한 부분을 더욱 강조하여 표현
  - 상세한 부분은 고주파 성분이므로, 저주파 성분을 제거하면 샤프닝 효과

−1	−1	−1
−1	9	−1
−1	−1	−1

(a) 샤프닝 회선 마스크 1

샤프닝 마스크1 적용

−1/9	−1/9	−1/9
−1/9	8/9	−1/9
−1/9	−1/9	−1/9

고주파 필터

고주파 마스크 적용

5. 경계선 검출

- 에지(edge)
  - 디지털 영상의 밝기가 높은 값에서 낮은 값으로 변하는 지점
  - 영상을 구성하는 객체 간의 경계(=경계선)
  - 물체 식별, 위치/모양/크기 등을 인지할 수 있는 정보 제공
- 간단한 에지 추출 기법
  - 연산 자체가 간단하고 빠름
  - 유사 연산자(Homogeneity Operator)와 차 연산자(Difference Operator)가 있음
- 미분을 이용한 에지 검출 방법
  - 에지가 화소의 밝기 변화율에 관여한다는 것
  - 1차 미분을 이용한 검출 방법과 2차 미분을 이용한 검출 방법이 있음

5. 경계선 검출

- 에지 검출기 : 화소간의 차이를 이용하는 것
  - 수평 에지 검출

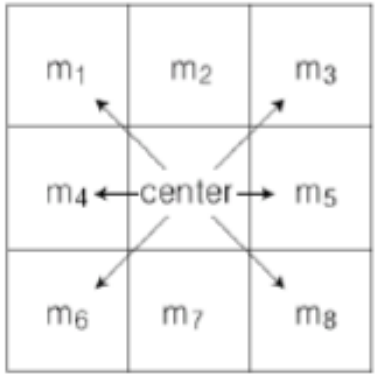
0	−1	0
0	1	0
0	0	0

(b) 수평 에지 검출 마스크

수평 에지 검출

5. 경계선 검출

- 유사 연산자
  - 가장 단순한 에지 검출 방법
  - 뺄셈연산이 여러 번 수행되어 계산 시간이 많이 소요

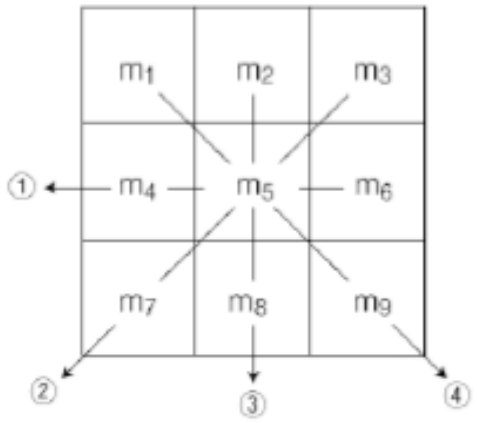


New Pixel = max(|center-m1|...|center-m8|)
총 8번 수행

유사 연산자 기법으로 검출된 에지

5. 경계선 검출

- 차 연산자
  - 유사 연산자의 계산 시간이 오래 걸리는 단점을 보안
  - 뺄셈 연산이 화소당 네 번으로 연산 시간이 빠름

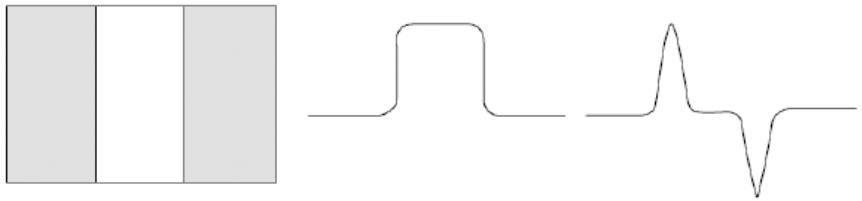


New Pixel = max(|m1-m9|...|m6-m4|)
총 4번 수행

차 연산자 기법으로 검출된 에지

5. 경계선 검출

- 영상의 에지는 화소의 밝기 값이 급격히 변하는 부분
- 함수의 변화분을 찾는 미분 연산이 이용됨



[그림 7-8] 영상의 밝기 변화와 미분 값 변화

- 1차 미분을 이용한 에지 검출
  - 좌표 (x,y)에서 각 방향으로의 편미분

5. 경계선 검출

- 1차 미분 회선 마스크
  - 종류가 다양함
  - 로버츠(Roberts), 소벨(Sobel), 프리윗(Prewitt) 마스크가 대표적
  - 각 회선 마스크의 고유한 특징이 있음.

[그림 7-10] 다양한 1차 미분 회선 마스크

5. 경계선 검출

- 소벨 마스크
  - 장점: 돌출된 값을 비교적 잘 평균화함.
  - 단점: 대각선 방향에 놓인 에지에 더 민감하게 반응함

원본 영상

수평 방향 에지

5. 경계선 검출

- 2차 미분을 이용한 에지 검출
  - 미분을 한번 더 수행하므로, 1차 미분의 단점(에지가 있는 영역을 지날 때 민감하게 반응)을 완화시킴
  - 장점: 검출된 에지를 끊거나 하지 않고 연결된 폐곡선을 형성함
  - 단점: 고립된 잡음에 민감하고, 윤곽의 강도만 검출하지 방향은 구하지 못함.

$$f'(x) = \frac{\partial f}{dx} = f(x+1) - f(x)$$

$$f''(x) = \frac{d^2 f}{dx^2} = \frac{df(x+1)}{dx} - \frac{df(x)}{dx} = (f(x+1) - f(x)) - (f(x) - f(x-1))$$

$$= f(x+1) - 2f(x) + f(x-1)$$

5. 경계선 검출

- 라플라시안(Laplacian) 에지 검출
  - 대표적인 2차 미분 연산자로, 에지 검출 성능이 우수함
  - 에지의 방향은 검출하지 못하고, 실제보다 많은 에지를 검출

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

라플라시안 회선 마스크
회선 마스크의 합은 0

라플라시안 회선 마스크로 검출된 에지 영상

5. 경계선 검출

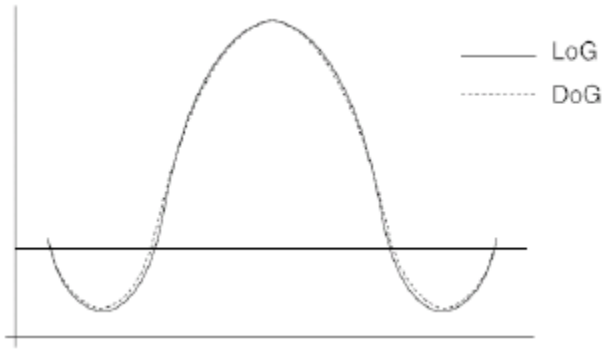
- LoG(Laplacian of Gaussian) 연산자
  - 잡음에 민감한 라플라시안의 문제를 해결하기 위해 만듦.
  - 계산 시간이 많이 소요됨

LoG 연산자 공식

$$\mathrm{LoG}(x,\ y) = \frac{1}{\pi\sigma^4}\left[1 - \frac{(x^2+y^2)}{2\sigma^2}\right] - e^{\frac{-(x^2+y^2)}{2\sigma^2}}$$

5. 경계선 검출

- DoG(Difference of Gaussians) 연산자
  - 계산 시간이 많이 걸리는 LoG의 단점 보완 위해 등장



LoG와 DoG 그래프 비교

0	0	−1	−1	−1	0	0
0	−2	−3	−3	−3	−2	0
−1	−3	5	5	5	−3	−1
−1	−3	5	16	5	−3	−1
−1	−3	5	5	5	−3	−1
0	−2	−3	−3	−3	−2	0
0	0	−1	−1	−1	0	0

7x7 DoG 마스크

DoG로 검출된 에지 영상

마무리

- 느낀 점
  - 영상 처리의 종류와 구현 방법에 대한 알게 됨
  - 라이브러리가 제공하는 기능을 low level로 구현함으로써 자신감을 얻음
- 한계점
  - User Interface 개선 필요
  - 컬러 이미지 영상 처리 불가능
- 발전 방향 구현 내용
  - C++, Python으로 구현
  - Ctrl+Z (취소하기) 기능 구현

감사합니다