

OpenCV 없이 C언어로 구현한 영상처리

[Intel] 엡지 AI SW 아카데미- 절차지향 프로그래밍

신나라

요약 | Overview

프로젝트 개요

프로젝트
세부사항

프로젝트
마무리

1

프로젝트 개요

목표

- ✓ OpenCV 라이브러리 사용 X
- ✓ 그에 준하는 기능을 구현할 것
- ✓ 편리할 것

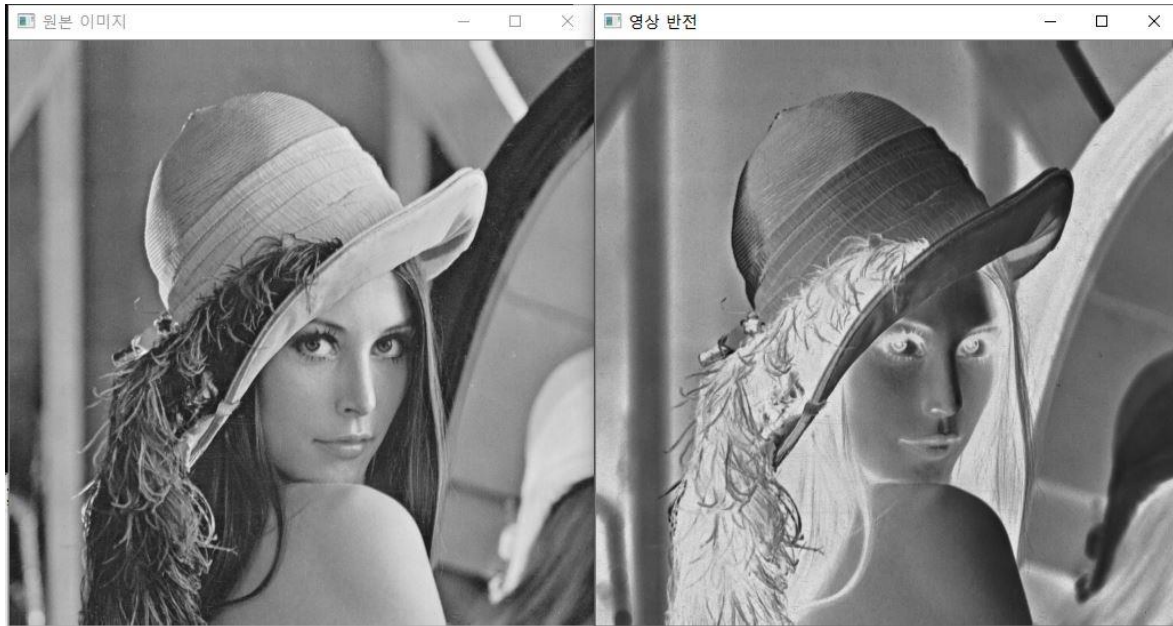
특이점

```
void freeInputMemory() {  
    if (inImage == NULL)  
        return;  
    for (int i = 0; i < inH; i++)  
        free(inImage[i]);  
    free(inImage);  
    inImage = NULL;  
}  
  
void mallocInputMemory() {  
    inImage = (unsigned char**)malloc(sizeof(unsigned char*) * inH);  
    for (int i = 0; i < inH; i++)  
        inImage[i] = (unsigned char*)malloc(sizeof(unsigned char) * inW);  
}
```

inImage와 outImage

→ 다음과 같이 메모리할당&해제 과정을 거침

계획



- ✓ OpenCV 없이 C언어로 영상처리
- ✓ 다양한 기능 구현
- ✓ 메뉴 선택 방식

2

프로젝트 세부사항

개발 환경

- ✓ 운영 체제 : Windows 11
- ✓ 개발 언어 : C
- ✓ 개발 도구 : Visual Studio 2022

구현 기능

- ✓ 화소점 처리 (13개)
- ✓ 기하학 처리 (9개)
- ✓ 히스토그램 처리 (9개)
- ✓ 경계선 검출 (12개)

✓ 파일 열기

```
D:\MyProject\W10일자Wx64W\ x + v

## GrayScale Image Processing (Beta 4) ##
0.열기 8.저장 9.종료
1.화소점 처리 2.기하학 처리 3.히스토그램 처리 4.경계선 검출
```

0 누르면




```
D:\MyProject\W10일자Wx64W\ x + v

## GrayScale Image Processing (Beta 4) ##
0.열기 8.저장 9.종료
1.화소점 처리 2.기하학 처리 3.히스토그램 처리 4.경계선 검출
파일명-->loopy_256
```

원하는 파일명 입력



```
226 strcat(fullName, tmpName);
227 strcat(fullName, ".raw");
228 strcpy(fileName, fullName);
229
230 // (중요!) 이미지의 폭과 높이를 결정
    b");
    해주세요.");
    ; // 파일의 끝으로 이동
    p); // 나 어디쯤이지? 262,144 --> 512
    기
    int4 = int4 * (int4/size);
    // 메모리 할당
```



```
## GrayScale Image Processing (Beta 4) ##
0.열기 8.저장 9.종료
1.화소점 처리 2.기하학 처리 3.히스토그램 처리 4.경계선 검출
```

Grayscale의 파일 열림

✓ 파일 열기

```
## GrayScale Image Processing (Beta 4) ##
0.열기 8.저장 9.종료
1.화소점 처리 2.기하학 처리 3.히스토그램 처리 4.경계선 검출
```

0 누르고



```
## Grayscale Image Processing (Beta 4) ##
0. 열기 8. 저장 9. 종료
1. 화소점 처리 2. 기하학 처리 3. 히스토그램 처리 4. 경계선 검출
파일명-->loopybabo
```

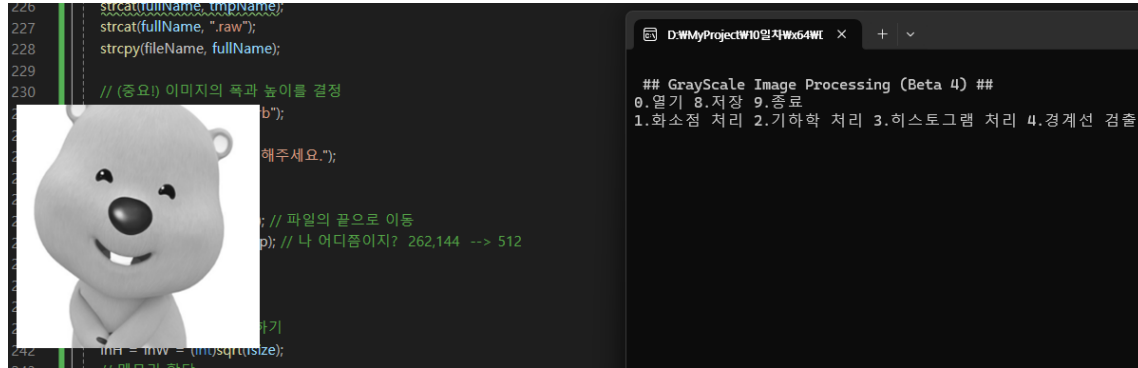
if 존재하지 않는 파일 입력 시



```
## GrayScale Image Processing (Beta 4) ##
0.열기 8.저장 9.종료
1.화소점 처리 2.기하학 처리 3.히스토그램 처리 4.경계선 검출
파일명-->loopybabo
파일명을 확인해주세요.
## GrayScale Image Processing (Beta 4) ##
0.열기 8.저장 9.종료
1.화소점 처리 2.기하학 처리 3.히스토그램 처리 4.경계선 검출
```

"파일명을 확인해주세요." 출력

✓ 화소점 처리



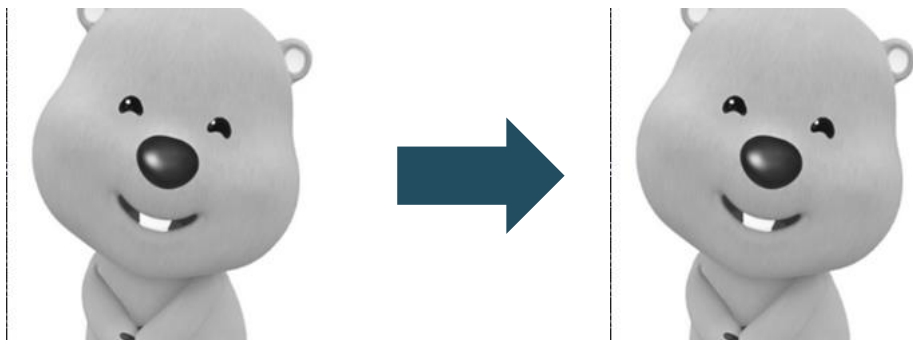
파일 열린 상태에서 1 누르면



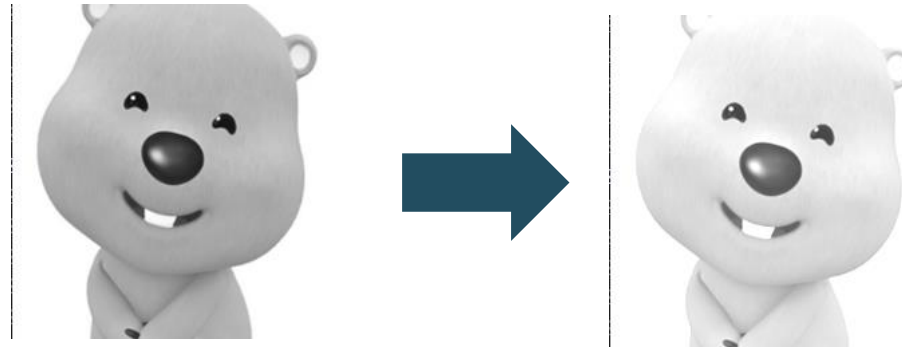
```
## GrayScale Image Processing (Beta 4) ##
8.저장 9.뒤로 가기
1.화소점 처리
A.동일 B.밝게 C.어둡게 D.반전 E.흑백 F.감마 G.CAP파라볼라 H.CUP파라볼라 I.흑백(평균값) J.흑백(중앙값) K.명암대비스트레칭
L.포스터라이징 M.범위강조
```

화소점 처리 기능 출력

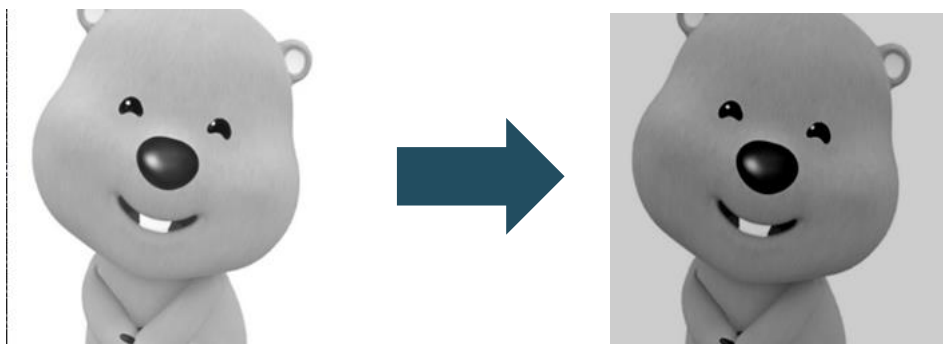
✓ 화소점 처리



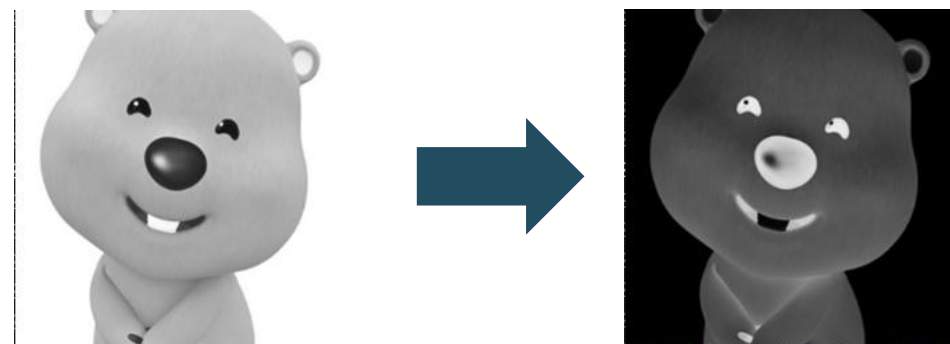
A. 동일
 $\text{outImage} = \text{inImage}$



B. 밝게
 $\text{outImage} = \text{inImage} + 50$



C. 어둡게
 $\text{outImage} = \text{inImage} - 50$



D. 반전
 $\text{outImage} = 255 - \text{inImage}$

✓ 화소점 처리



E.흑백

inImage > 127 이면 outImage = 255
아니면 outImage = 0



F.감마 (1.5)

outImage = $255 * (inImage / 255)^{(1 / gamma)}$



G.CAP파라볼라

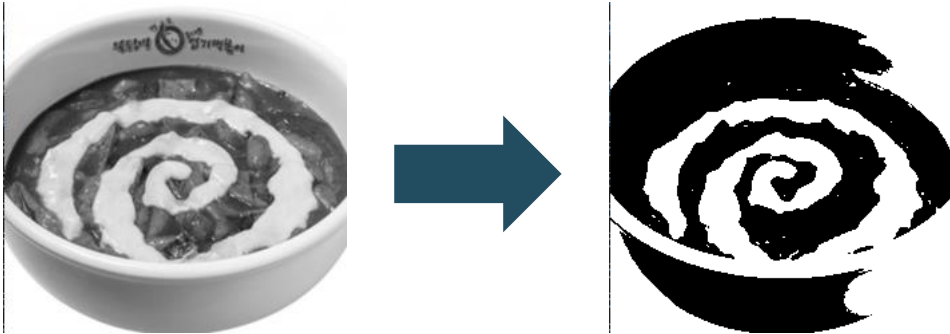
outImage = $255 * (inImage / 127 - 1)^2$



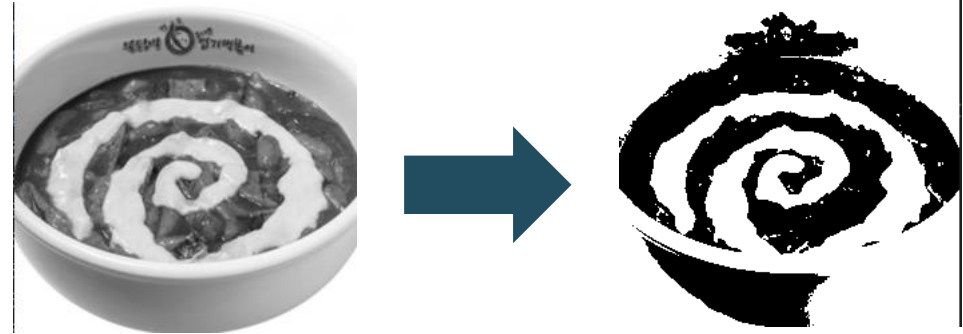
H.CUP파라볼라

outImage = $255 - 255 * (inImage / 127 - 1)^2$

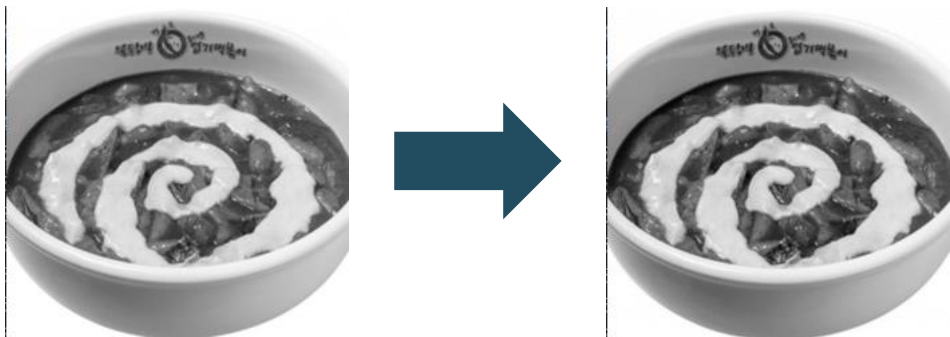
✓ 화소점 처리



I.흑백(평균값)

$$\text{hap} += \text{inImage}; \text{avg} = \text{hap} / (\text{inH} * \text{inW})$$


J.흑백(중앙값)

$$\text{arr} = \text{inImage}; \text{qsort}(\text{arr}); \text{mid} = \text{arr}[(\text{inH} * \text{inW}) / 2]$$


K.명암대비스트레칭

$$\text{outImage} = \text{tmp}[\text{inImage}]$$

$$\text{tmp} = (i - \text{min}) * (255 / (\text{max} - \text{min})) \rightarrow i++ \sim \text{max}$$


L.포스터라이징

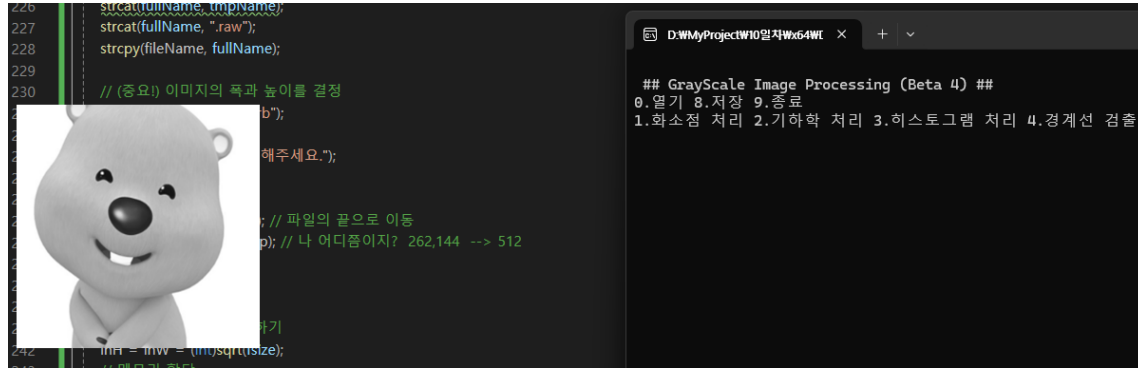
$$0 < \text{inImage} \leq 32 \text{ 이면 } \text{outImage} = 16 \dots$$

✓ 화소점 처리

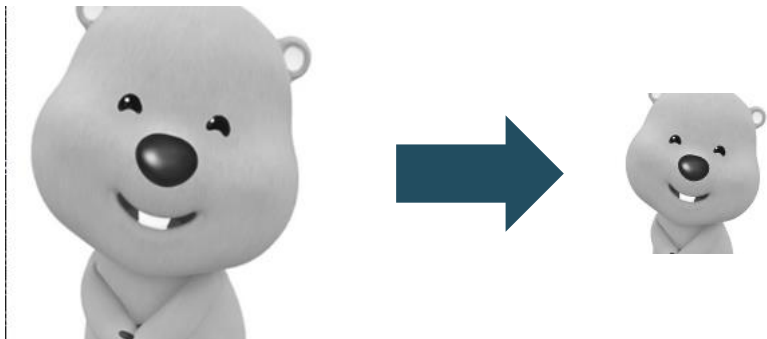


M.범위 강조(100~150)
시작값 \leq inImage < 끝값 이면 outImage = 255
아니면 outImage = 0

✓ 기하학 처리



✓ 기하학 처리



A. 축소

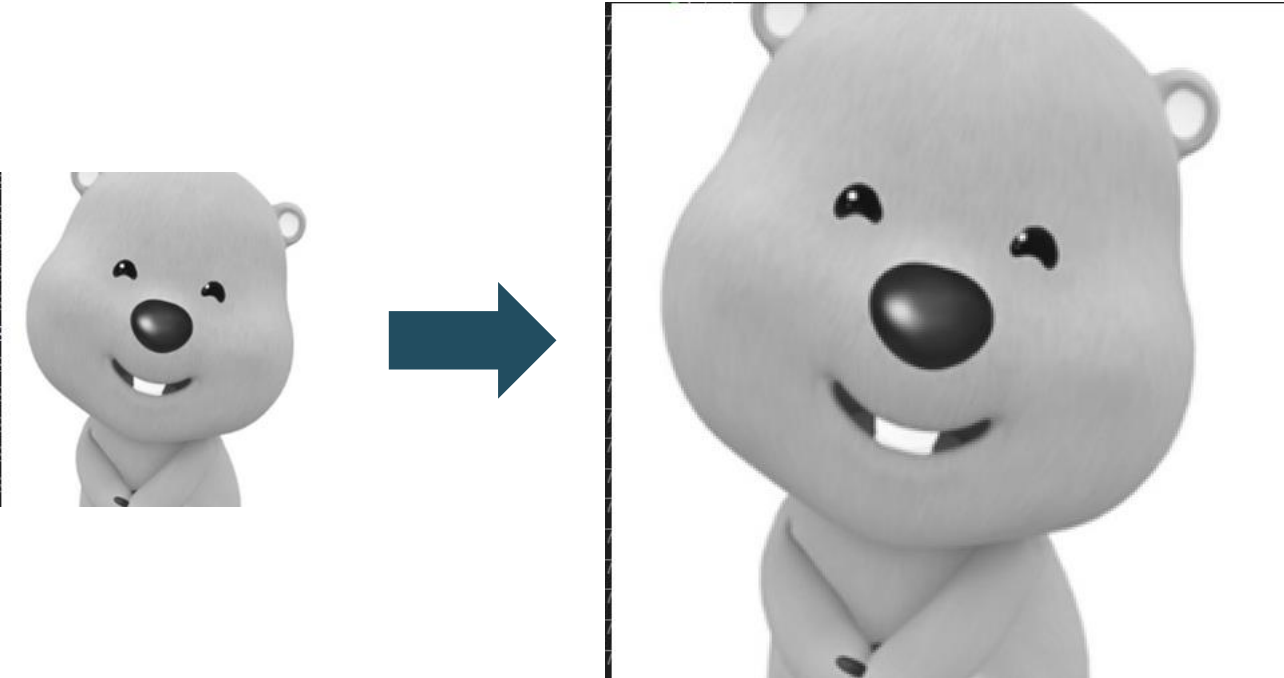
$$\text{outImage}[i/\text{scale}][k/\text{scale}] = \text{inImage}$$



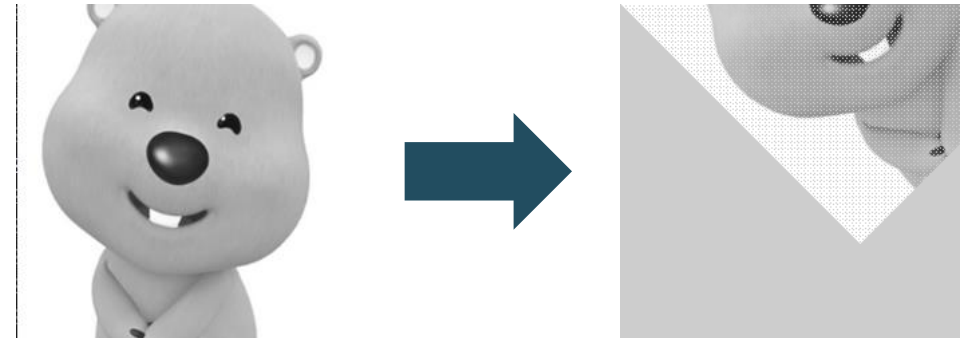
B. 확대(포워딩)

$$\text{outImage}[i*\text{scale}][k*\text{scale}] = \text{inImage}$$

✓ 기하학 처리



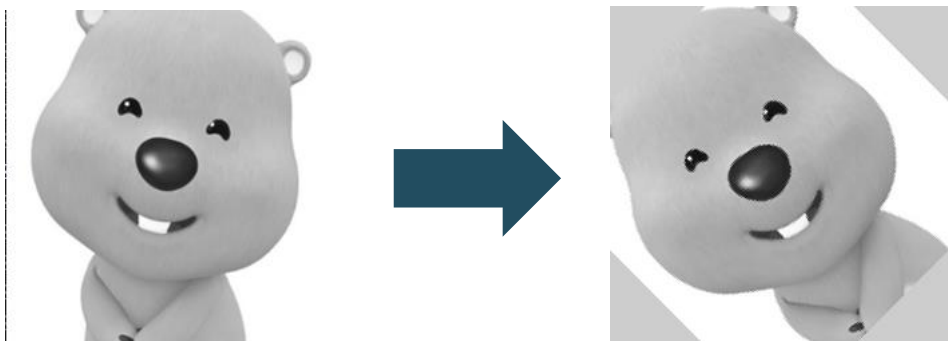
C.확대(백워딩)

$$\text{outImage} = \text{inImage} [i/\text{scale}][k/\text{scale}]$$


D.회전(기본)

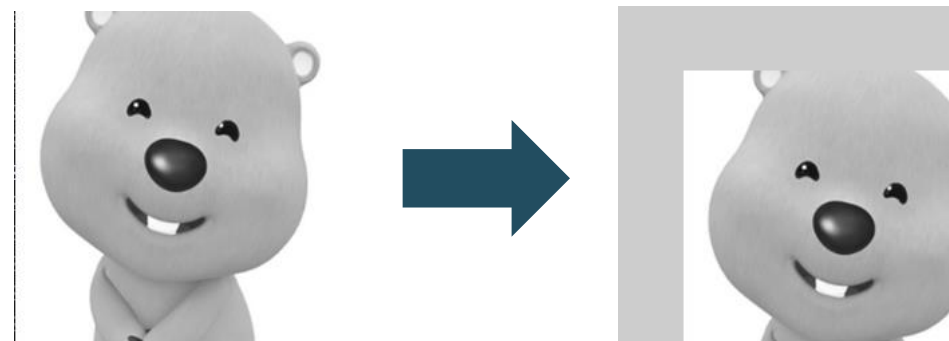
$$0 \leq x_d < \text{outH} \text{ 이고 } 0 \leq y_d < \text{outW} \text{ 이면}$$
$$\text{outImage}[x_d][y_d] = \text{inImage}[x_s][y_s];$$

✓ 기하학 처리



E.회전(중양, 백워딩)

$0 \leq xs < outH$ 이고 $0 \leq ys < outW$ 이면
 $outImage[xd][yd] = inImage[xs][ys];$



F.이미지 이동

$0 \leq x + k < inH$ 이고 $0 \leq y + i < inW$ 이면
 $outImage[i + y][k + x] = inImage[i][k];$

✓ 기하학 처리



G.좌우반전

$$\text{outImage}[i][k] = \text{inImage}[i][\text{inW} - k - 1]$$



H.상하반전

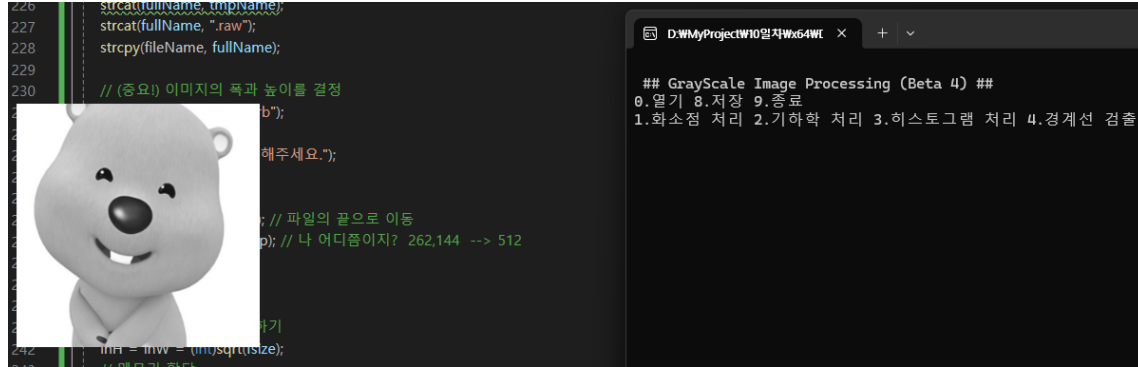
$$\text{outImage}[i][k] = \text{inImage}[\text{inH} - 1 - i][k]$$



H.좌우상하반전

$$\text{outImage}[i][k] = \text{inImage}[\text{inH} - 1 - i][\text{inW} - k - 1]$$

✓ 히스토그램 처리



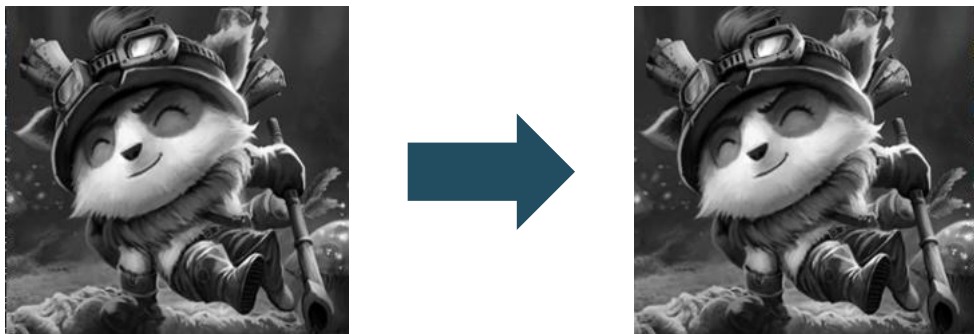
파일 열린 상태에서 3 누르면



```
## GrayScale Image Processing (Beta 4) ##  
8. 저장 9. 뒤로 가기  
3. 히스토그램 처리  
A. 히스토그램 스트레칭 B. 엔드-인 C. 평활화 D. 엠보싱 E. 블러 F. 샤프닝 G. 고주파필터 샤프닝 H. 저주파필터 샤프닝 I. 가우시안 블러  
무딩
```

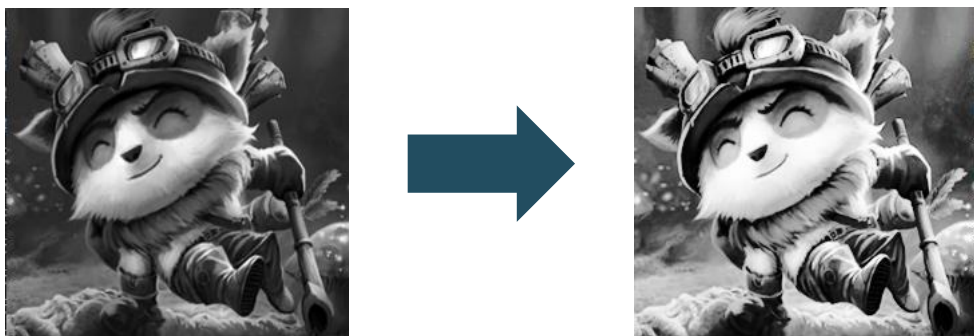
히스토그램 처리 기능 출력

✓ 히스토그램 처리



A.히스토그램 스트레칭

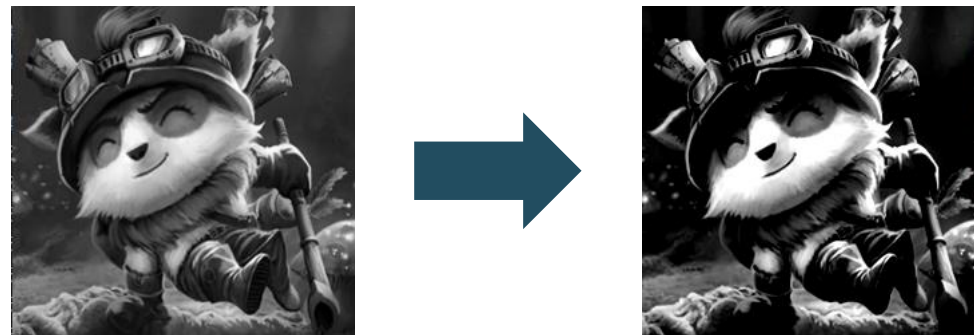
$$\text{outImage} = (\text{old} - \text{low}) / (\text{high} - \text{low}) * 255)$$



C.평활화

$$\text{outImage} = \text{normalHisto}[\text{inImage}]$$

normalHisto = 정규화된 히스토그램



B.엔드-인

$$\text{outImage} = (\text{old} - \text{low}) / (\text{high} - \text{low}) * 255)$$

$$\text{high} -= 50 \quad \text{low} += 50$$



D.엠보싱

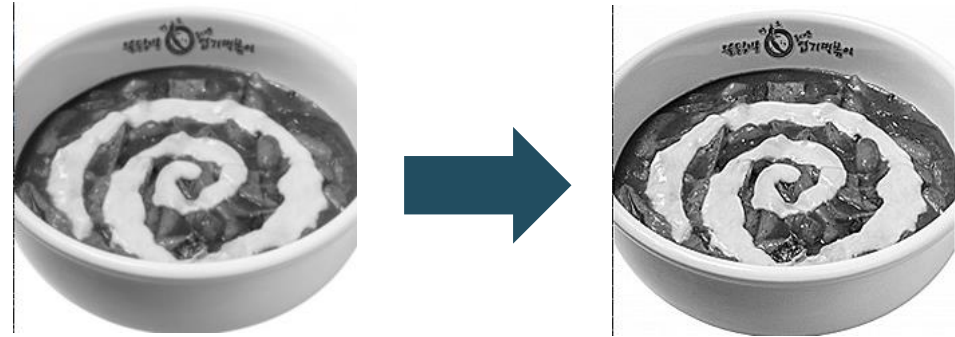
$$\text{mask} = \{-1, 0, 0\}, \{0, 0, 0\}, \{0, 0, 1\}$$

$$\text{outImage} = \text{tmpInImage} * \text{mask}$$

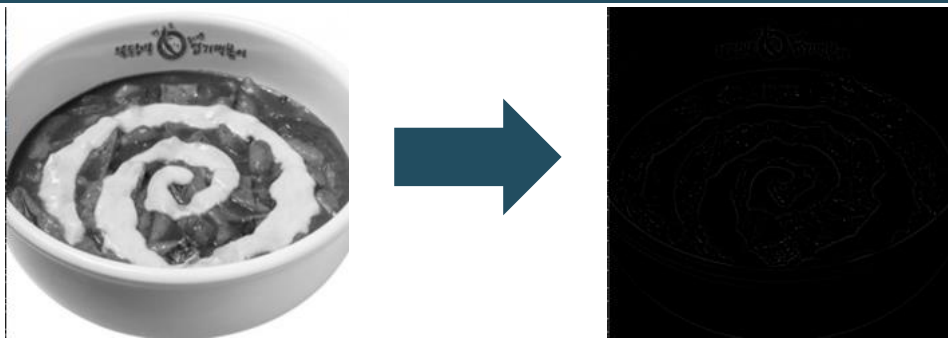
✓ 히스토그램 처리



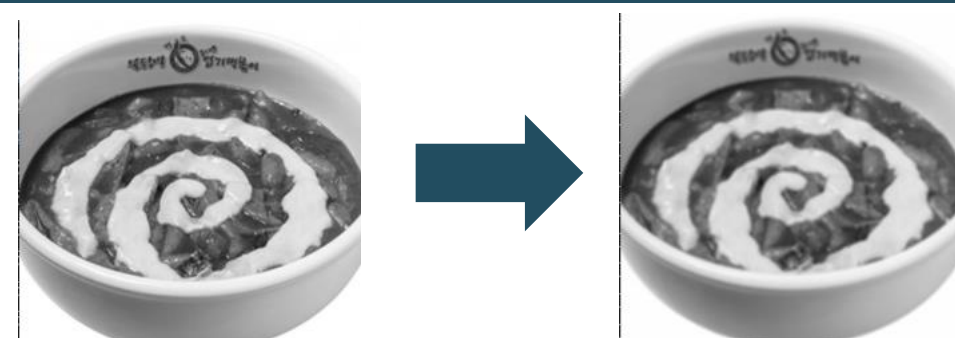
E.블러

$$\text{mask} = \left\{ \left\{ \frac{1}{9}, \frac{1}{9}, \frac{1}{9} \right\}, \left\{ \frac{1}{9}, \frac{1}{9}, \frac{1}{9} \right\}, \left\{ \frac{1}{9}, \frac{1}{9}, \frac{1}{9} \right\} \right\}$$


F.샤프닝

$$\text{mask} = \left\{ \left\{ 0, -1, 0 \right\}, \left\{ -1, 5, -1 \right\}, \left\{ 0, -1, 0 \right\} \right\}$$


G.고주파필터 샤프닝

$$\text{mask} = \left\{ \left\{ -\frac{1}{9}, -\frac{1}{9}, -\frac{1}{9} \right\}, \left\{ -\frac{1}{9}, \frac{8}{9}, -\frac{1}{9} \right\}, \left\{ -\frac{1}{9}, -\frac{1}{9}, -\frac{1}{9} \right\} \right\}$$


H.저주파필터 샤프닝

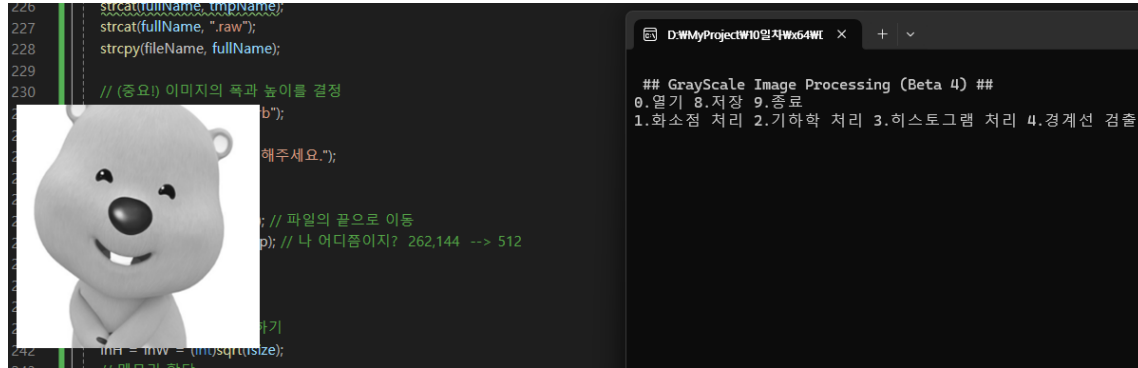
$$\text{mask} = \left\{ \left\{ \frac{1}{9}, \frac{1}{9}, \frac{1}{9} \right\}, \left\{ \frac{1}{9}, \frac{1}{9}, \frac{1}{9} \right\}, \left\{ \frac{1}{9}, \frac{1}{9}, \frac{1}{9} \right\} \right\}$$

✓ 히스토그램 처리



I.가우시안 스무딩
mask = $\{\{1/16, 1/8, 1/16\},$
 $\{1/8, 1/4, 1/8\},$
 $\{1/16, 1/8, 1/16\}\}$

✓ 경계선 검출



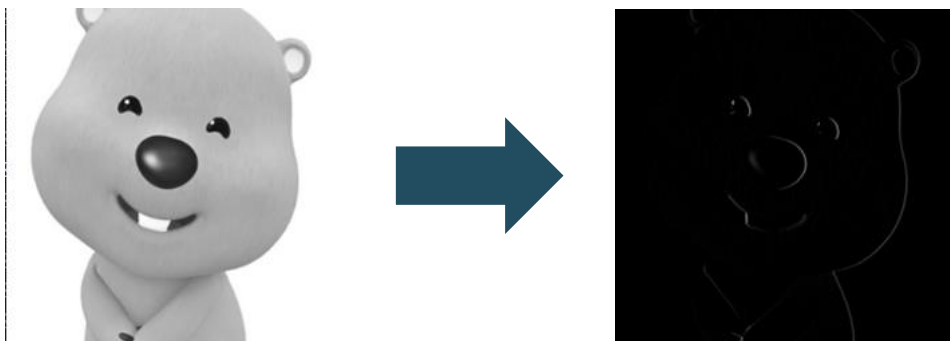
파일 열린 상태에서 4 누르면



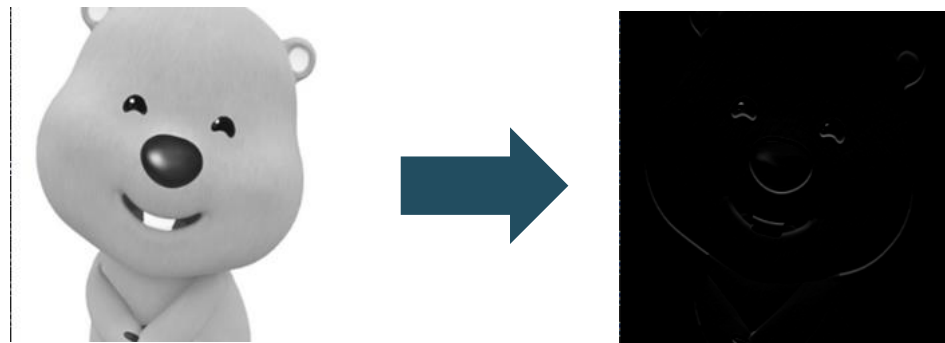
```
## GrayScale Image Processing (Beta 4) ##  
8.저장 9.뒤로 가기  
4.경계선 검출  
A.경계선1(수직검출) B.경계선2(수평검출) C.경계선3(유사연산자) D.로버츠(행 검출) E.로버츠(열 검출) F.소벨(행 검출) G.소벨  
(열 검출) H.프리윗(행 검출) I.프리윗(열 검출) J.라플라시안(1) K.라플라시안(2) L.라플라시안(3)
```

경계선 검출 기능 출력

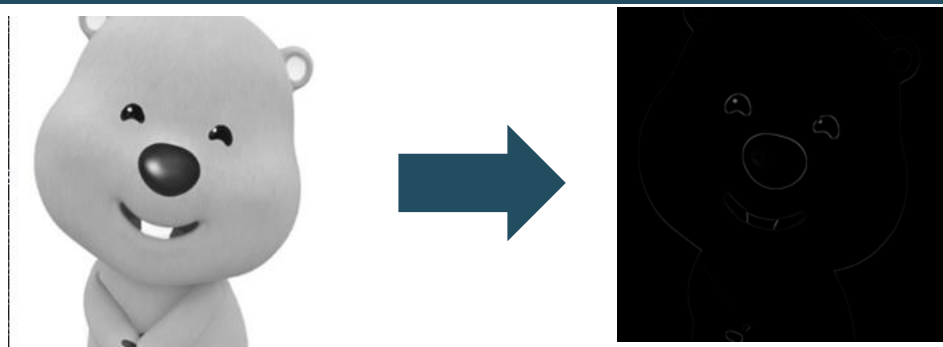
✓ 경계선 검출



A.경계선1(수직검출)

$$\text{mask} = \{\{0, 0, 0\}, \{-1, 1, 0\}, \{0, 0, 0\}\}$$
$$\text{outImage} = \text{tmpInImage} * \text{mask}$$


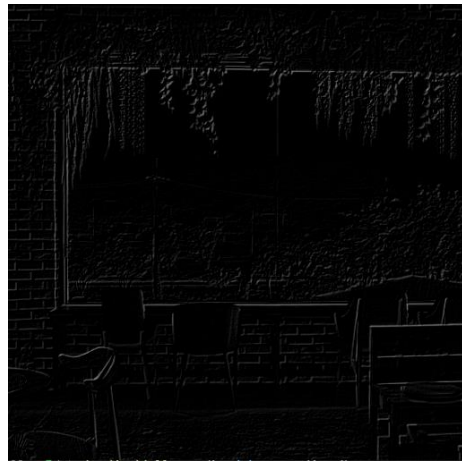
B.경계선2(수평검출)

$$\text{mask} = \{\{0, -1, 0\}, \{0, 1, 0\}, \{0, 0, 0\}\}$$


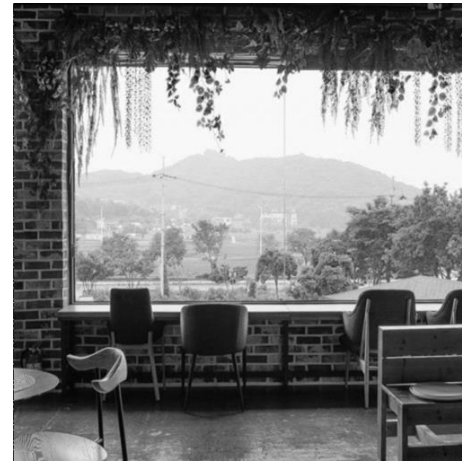
C.경계선3(유사연산자)

$$\text{mask} = \{\{1, 1, 1\}, \{1, 1, 1\}, \{1, 1, 1\}\}$$
$$\text{outImage} = (\text{tmpInImage}[i + 1][k + 1] - \text{tmpInImage}[i + m][k + n]) * (\text{tmpOutImage}/255)$$

✓ 경계선 검출



D.로버츠(행검출)
mask = $\begin{Bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{Bmatrix}$

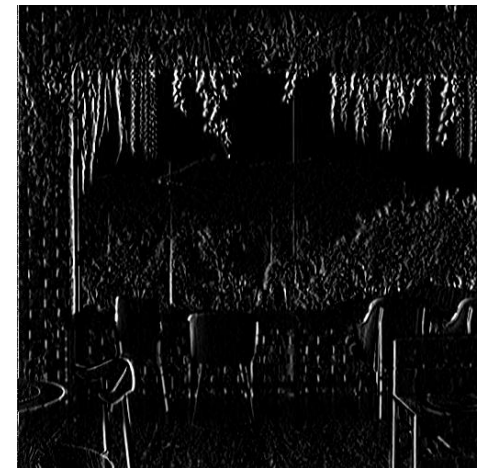


E.로버츠(열검출)
mask = $\begin{Bmatrix} 0 & 0 & -1 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{Bmatrix}$

✓ 경계선 검출

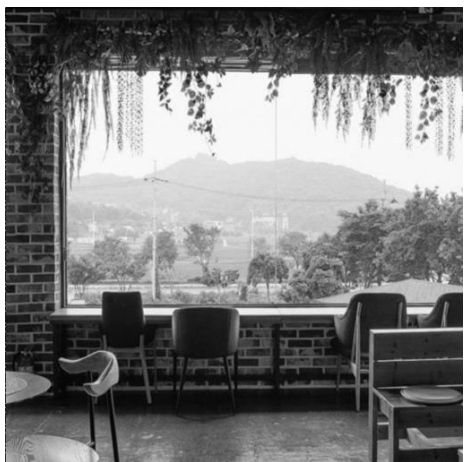


F.소벨(행검출)
mask = $\begin{Bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{Bmatrix}$

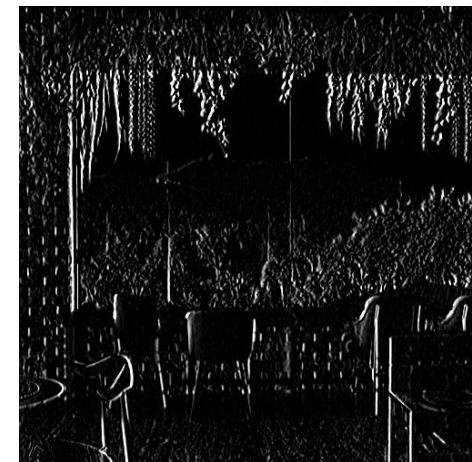


G.소벨(열검출)
mask = $\begin{Bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{Bmatrix}$

✓ 경계선 검출



H.프리윗(행검출)
mask = $\begin{Bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{Bmatrix}$



I.프리윗(열검출)
mask = $\begin{Bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{Bmatrix}$

✓ 경계선 검출



J. 라플라시안(1)

$$\text{mask} = \{\{0, -1, 0\}, \{-1, 4, -1\}, \{0, -1, 0\}\}$$


K. 라플라시안(2)

$$\text{mask} = \{\{1, 1, 1\}, \{1, -8, 1\}, \{1, 1, 1\}\}$$


L. 라플라시안(3)

$$\text{mask} = \{\{-1, -1, -1\}, \{-1, 8, -1\}, \{-1, -1, -1\}\}$$

✓ 파일 저장

```
## GrayScale Image Processing (Beta 4) ##  
8.저장 9.뒤로 가기  
4.경계선 검출  
A.경계선1(수직검출) B.경계선2(수평검출) C.경계선3(유사연산자) D.로버츠(행 검출) E.로버츠(열 검출) F.소벨(행 검출) G.소벨  
(열 검출) H.프리윗(행 검출) I.프리윗(열 검출) J.라플라시안(1) K.라플라시안(2) L.라플라시안(3)
```

8 누르고



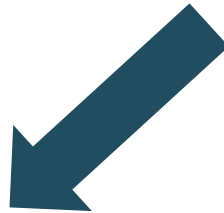
```
## GrayScale Image Processing (Beta 4) ##  
8.저장 9.뒤로 가기  
4.경계선 검출  
A.경계선1(수직검출) B.경계선2(수평검출) C.경계선3(유사연산자) D.로버츠(행 검출) E.로버츠(열 검출) F.소벨(행 검출) G.소벨  
(열 검출) H.프리윗(행 검출) I.프리윗(열 검출) J.라플라시안(1) K.라플라시안(2) L.라플라시안(3)  
파일명-->out100  
저장 완료
```

원하는 파일명 입력 후 엔터 → 저장 완료

✓ 프로그램 종료

```
## GrayScale Image Processing (Beta 4) ##  
8.저장 9.뒤로 가기  
4.경계선 검출  
A.경계선1(수직검출) B.경계선2(수평검출) C.경계선3(유사연산자) D.로버츠(행 검출) E.로버츠(열 검출) F.소벨(행 검출) G.소벨  
(열 검출) H.프리윗(행 검출) I.프리윗(열 검출) J.라플라시안(1) K.라플라시안(2) L.라플라시안(3)
```

9 누르면 → 뒤로 가짐



```
D:\MyProject\10일차\wx64\W... x + v  
  
## GrayScale Image Processing (Beta 4) ##  
0.열기 8.저장 9.종료  
1.화소점 처리 2.기하학 처리 3.히스토그램 처리 4.경계선 검출
```

다시 9를 누르면



```
## GrayScale Image Processing (Beta 4) ##  
0.열기 8.저장 9.종료  
1.화소점 처리 2.기하학 처리 3.히스토그램 처리 4.경계선 검출  
  
D:\MyProject\10일차\wx64\Debug\10일차.exe(프로세스 12296개)이(가) 종료되었습니다(코드: 0개).  
이 창을 닫으려면 아무 키나 누르세요...
```

프로그램 종료

3

프로젝트 마무리

느낀 점

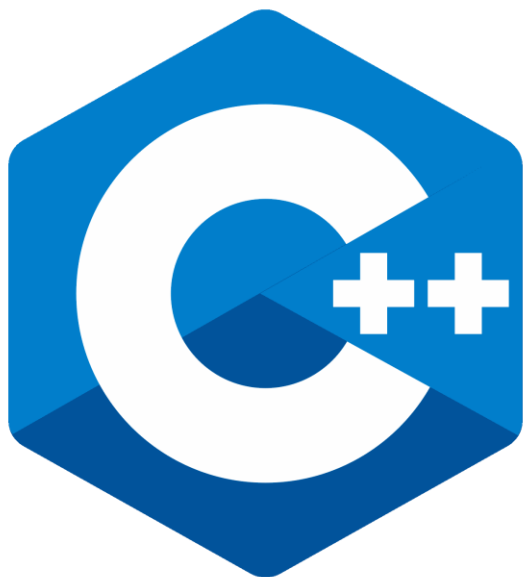
- ✓ 라이브러리 개발자들이 대단하다는 생각이 들었음.
- ✓ 직접 다양한 기능들을 구현해 본 좋은 경험이 쌓인 듯함.

한계점

```
void freeInputMemory() {  
    if (inImage == NULL)  
        return;  
    for (int i = 0; i < inH; i++)  
        free(inImage[i]);  
    free(inImage);  
    inImage = NULL;  
}  
  
void mallocInputMemory() {  
    inImage = (unsigned char**)malloc(sizeof(unsigned char*) * inH);  
    for (int i = 0; i < inH; i++)  
        inImage[i] = (unsigned char*)malloc(sizeof(unsigned char) * inW);  
}
```

다음과 같이 메모리 할당&해제 과정을 거치지 않으면 원하는 기능을 모두 구현하기는 어려움

발전 방향



C언어로 구현해본 영상처리를 C++과 Python으로도 구현해볼 계획이 있음.

감사합니다 😊

[Github : sournara/intel_SW_Edge_AI_Academy \(github.com\)](https://github.com/sournara/intel_SW_Edge_AI_Academy)