**영상처리**

**3장 과제 #1**

****

|  |  |  |  |
| --- | --- | --- | --- |
| **학과/분반** | 컴퓨터공학과 / 02 | **학번** | 20210262 |
| **이름** | 김우혁 | **제출일** | 2025. 05. 01 (목) |
| **주제** | 관심 영역의 밝기 반전된 영상 생성하기 | | |

**■ 목차**

**• 실습 목표**

**• 해결 과정 및 주요 코드 설명**

**• 실행 결과**

**• 느낀점**

**■ 실습 목표**

기본 설정: 입력 영상의 시작위치는 (0, 0), 끝 위치는 (150, 100) 영역에 대한 밝기 반전

사용자 지정 설정: 입력 영상의 사용자가 지정한 시작위치, 끝 위치 영역에 대한 밝기 반전

**■ 해결 과정 및 주요 코드 설명**

- invert\_image\_m1(img, rect)

1. 원본 영상의 복제본을 만들어 사용한다.

|  |
| --- |
| dst = img.copy() |

2. 영상 유형의 구분은 영상 데이터의 shape 속성을 사용,

배열 인덱스를 사용하여 개별 픽셀에 접근하여 밝기 반전 연산을 수행한다.

|  |
| --- |
| img\_len = len(image.shape)  # 컬러 영상인 경우 채널 수가 3인 것을 확인  if img\_len == 3:  for x in range(start\_x, end\_x):  for y in range(start\_y, end\_y):  dst[y, x, 2] = 255 - dst[y, x, 2] # Red 채널  dst[y, x, 1] = 255 - dst[y, x, 1] # Green 채널  dst[y, x, 0] = 255 - dst[y, x, 0] # Blue 채널  return dst  elif img\_len == 2: # 그레이스케일 영상의 경우  for x in range(start\_x, end\_x):  for y in range(start\_y, end\_y):  dst[y, x] = 255 - dst[y, x]  return dst |

- invert\_image\_m2(img, rect)

1. 원본 영상의 복제본을 만들어 사용한다.

|  |
| --- |
| dst = img.copy() |

2. Numpy의 슬라이싱 연산과 브로드캐스팅 기법을 사용하여 밝기 반전 영산을 수행한다.

|  |
| --- |
| # 원하는 값들 뽑아내기  start\_x = rect[0]  start\_y = rect[1]  end\_x = rect[2]  end\_y = rect[3]  dst[start\_y:end\_y, start\_x:end\_x] = 255 - dst[start\_y:end\_y, start\_x:end\_x]  return dst |

**■ 실행 결과**

|  |
| --- |
| python invert\_ROI.py –image nature.jpg |
|  |

|  |
| --- |
| python invert\_ROI.py –image nature.jpg -s 50 10 -e 200 150 |
|  |
| python invert\_ROI.py --image nature\_grayscale.jpg | |
|  | |

|  |
| --- |
| python invert\_ROI.py --image nature\_grayscale.jpg -s 100 100 -e 300 400 |
|  |

**■ 느낀점**

이번 실습을 통해서 파이썬은 반복문이 매우 느리다는 것을 알게 되었고,

왠만하면 영상처리를 할 때에는 Numpy의 슬라이싱 연산을 잘 활용해야겠다고 느꼈다.