연습문제 7장

1. 입력 영상의 화소와 마스크의 구성으로 회선 알고리즘을 상세히 설명하시오.

|  |
| --- |
| Convolution은 마스크 내의 원소값과 공간 영역에 있는 입력 영상의 화소값들을 대응되게 곱하여 출력 화소값을 계산하는 것을 말한다. |

2. 블러링과 샤프닝을 비교하시오..

|  |
| --- |
| 블러링은 회선 마스크의 원소를 모두 같은 값으로 지정해 수행하며, 전체 합이 1이 되어야한다. 출력 영상에서 이웃하는 화소들이 비슷한 값을 갖기 때문에 부드러운 영상이 되며, 흐려지는 결과가 발생한다.  샤프닝은 회선 마스크에서 중심 계수와 주변 계수의 차이를 크게 만들어 출력 화소가 도드라지게 함으로써 선명하고 날카로운 영상을 만드는 방법이다. 중심 계수는 아주 큰 양수값을 갖게 하며, 주변 계수는 음수값을 갖게 해서 전체 합이 1이 되게 한다. |

3. 에지 검출 방법의 종류를 아는 데로 적으시오..

|  |
| --- |
| (1) 로버츠  (2) 소벨  (3) 프리윗  (4) 라플라시안  (5) LoG  (6) DoG  (7) Canny |

4. 대표적인 1차 미분 마스크의 종류를 적고, 그 장단점을 기술하시오.

|  |
| --- |
| 로버츠, 소벨, 프리윗  pros, 점진적으로 변화하는 부분도 검출.  cons, 너무 민감. |

5. 2차 미분 마스크의 종류와 특징에 대하여 설명하시오.

|  |
| --- |
| 라플라시안, LoG, Dog  1차 미분한 결과에 미분을 한 번 더 한다. |

6. 캐니 에지 알고리즘의 과정을 설명하시오.

|  |
| --- |
| (1) 블러링을 통한 노이즈 제거  (2) 화소 기울기의 강도와 방향 검출  (3) 비 최대치 억제  (4) 이력 임계값으로 에지 결정 |

8. 평균값 필터링과 미디언 필터링을 비교설명하시오.

|  |
| --- |
| 평균값 필터링은 마스크 범위의 입력 화소들을 평균하여 출력 화소를 결정. 블러링과 같은 효과가 난다. 미디언 필터링은 마스크 범위의 입력 화소들을 정렬하여 중간값을 출력화소로 결정한다. 임펄스 잡음이나 소금 후추 잡음은 마스크 범위 내에서 가장 큰 값 혹은 가장 작은 값이 되기 때문에 출력 화소에서 배제된다. 따라서 이와 같은 잡음 제거에 효과적이다. |

9. 모폴로지의 연산 종류에 대해서 적고, 각각에 대해서 설명하시오.

|  |
| --- |
| 침식 연산 : 객체의 크기가 축소되기 때문에 영상 내에 존재하는 작은 크기의 잡음을 제거하는데 효과적  팽창 연산 : 객체의 크기가 확대되어 객체 내부의 빈 공간을 메우는 역할 |

10. 저자가 구현한 Common.filters.filter() 함수는 명암도 영상에서 필터링을 수행한다. filter() 함수를 이용해서 컬러 영상에서 블러링과 샤프닝을 구현해 보자.

- 가우시안 블러링 혹은 평균 필터링으로 블러링 수행한다.

- 샤프닝은 3 x 3 크기의 마스크를 적용한다.

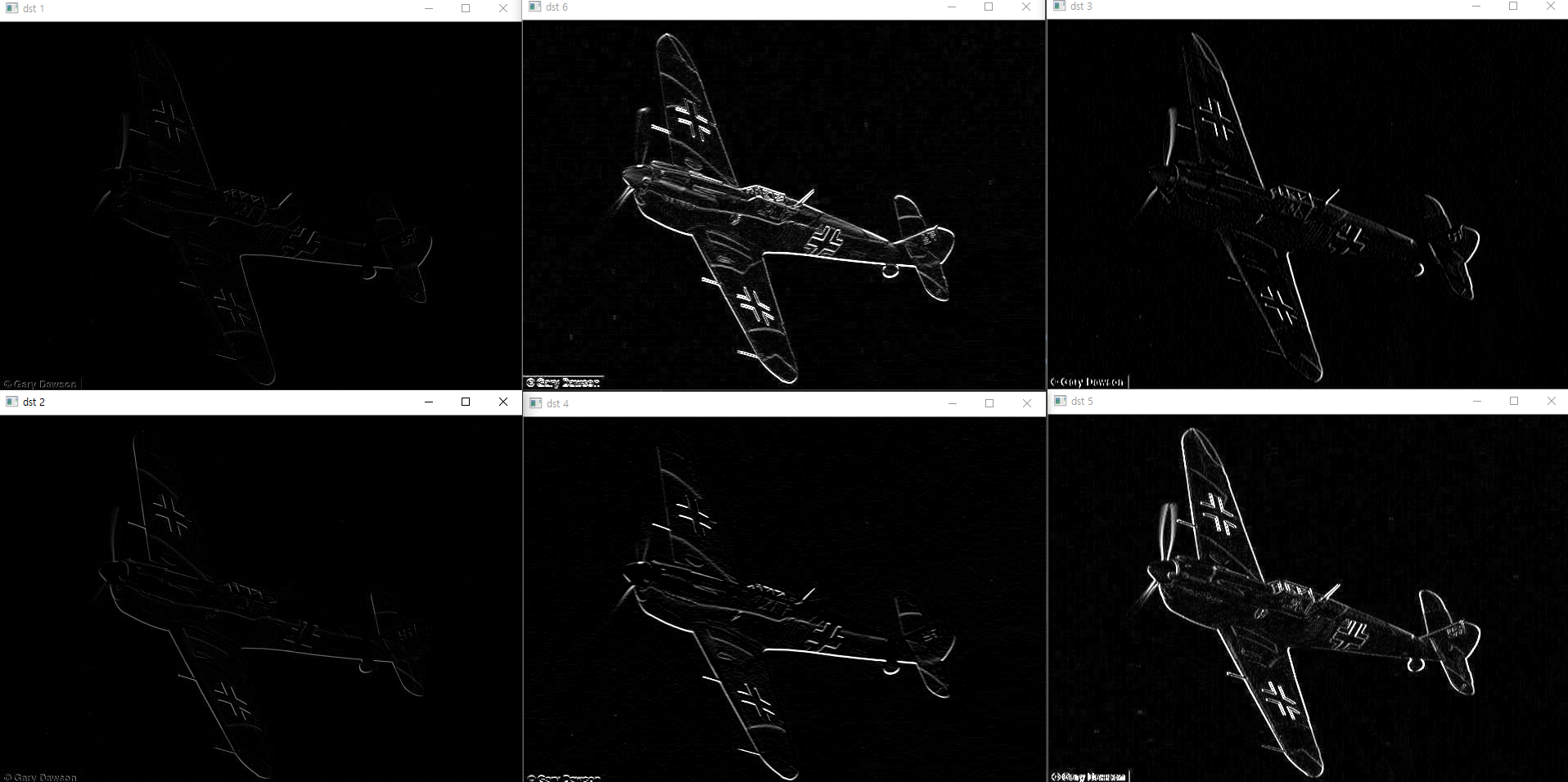
- 컬러 영상의 채널을 분리해서 각 채널에 블러링과 샤프닝을 수행하고, 다시 합쳐보자.

- 컬러 영상에서 바로 OpenCV 함수인 cv2.filter2D()를 적용해보자.



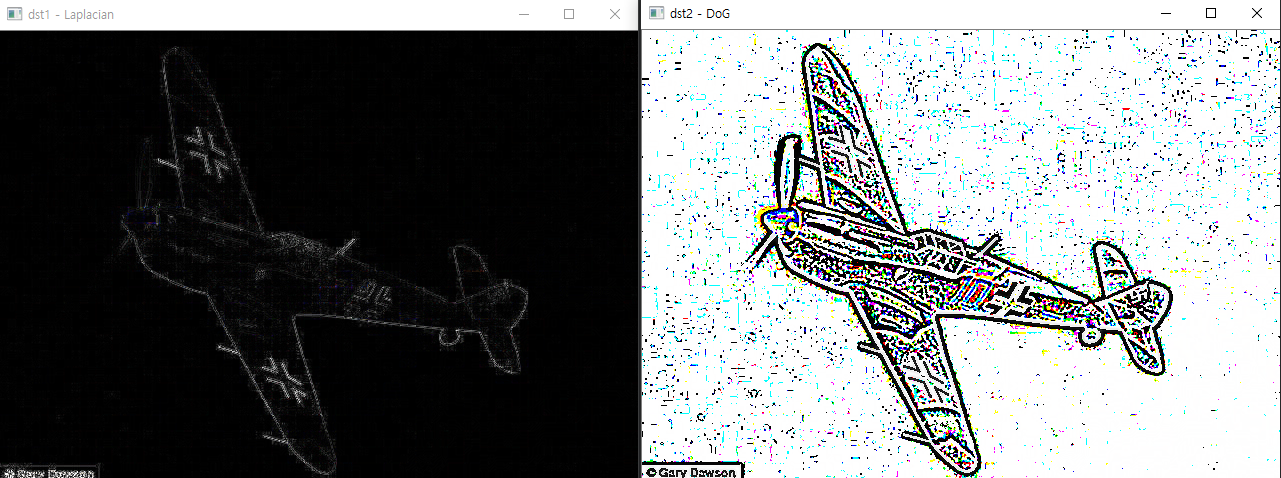
|  |
| --- |
| from Common.filters import filter import cv2, numpy as np  image = cv2.imread("../images/aircraft.jpg") if image is None:  raise Exception("영상파일 읽기 오류") b, g, r = cv2.split(image) bmask = np.array([1/9] \* 9, np.float32).reshape(3, 3) data = [0, -1, 0, -1, 5, -1, 0, -1, 0] smask = np.array(data, np.float32) for x in [b, g, r]:  x = filter(x, bmask)  x = filter(x, smask)  dst = cv2.merge((b, g, r)) dst2 = cv2.filter2D(image, -1, bmask) dst2 = cv2.filter2D(dst2, -1, smask) cv2.imshow("image", dst) cv2.imshow("dst2", dst2) cv2.waitKey(0) |

11. 1차 미분 연산을 수행하도록 마스크를 생성하여 직접 회선을 수행하시오(3가지 연산 마스크 적용).



|  |
| --- |
| import cv2, numpy as np from Common.filters import filter  image = cv2.imread("../images/aircraft.jpg", cv2.IMREAD\_GRAYSCALE) if image is None:  raise Exception("영상파일 읽기 오류")  dst = []  # Roberts Mask mask1 = [-1, 0, 0, 0, 1, 0, 0, 0, 0] mask2 = [0, 0, -1, 0, 1, 0, 0, 0, 0]  # Prewitt Mask mask3 = [-1, 0, 1, -1, 0, 1, -1, 0, 1] mask4 = [-1, -1, -1, 0, 0, 0, 1, 1, 1]  for mask in [mask1, mask2, mask3, mask4]:  mask = np.float32(mask).reshape(3, 3)  dst.append(filter(image, mask))  # Sobel Mask dst.append(cv2.convertScaleAbs(cv2.Sobel(np.float32(image), cv2.CV\_32F, 1, 0, 3))) dst.append(cv2.convertScaleAbs(cv2.Sobel(np.float32(image), cv2.CV\_32F, 0, 1, 3)))   for i, x in enumerate(dst):  cv2.imshow(f"dst {i + 1}", x) cv2.waitKey(0) |

12. 2차 미분 연산을 수행하는 함수로 에지 검출을 수행하시오(2가지 함수 이상).



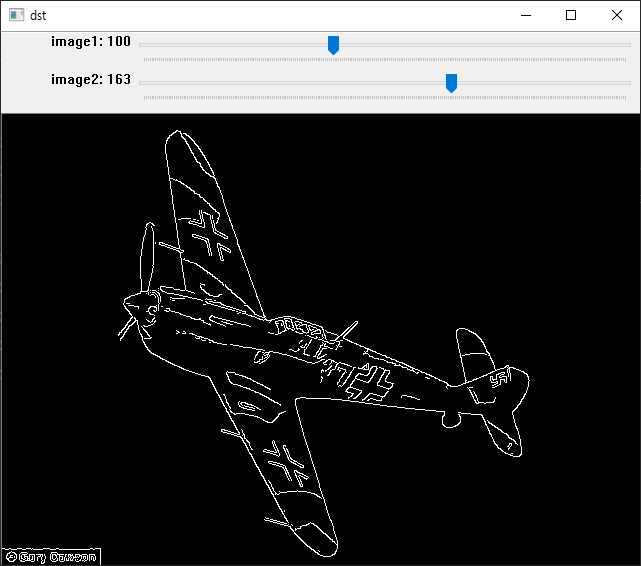
|  |
| --- |
| import cv2 import numpy as np  image = cv2.imread("../images/aircraft.jpg") if image is None:  raise Exception  dst1 = cv2.convertScaleAbs(cv2.Laplacian(image, cv2.CV\_32F, 1))  gaus1 = cv2.GaussianBlur(image, (3, 3), 0) gaus2 = cv2.GaussianBlur(image, (9, 9), 0) dst2 = gaus1 - gaus2  cv2.imshow("dst1 - Laplacian", dst1) cv2.imshow("dst2 - DoG", dst2) cv2.waitKey(0) |

13. 각기 다른 함수로 블러링을 수행하도록 프로그램을 작성하시오(3가지 함수 이상)



|  |
| --- |
| import cv2, numpy as np  image = cv2.imread("../images/aircraft.jpg") if image is None:  raise Exception("영상파일 읽기 오류")  blur1 = cv2.blur(image, (3, 3)) # 평균 블러링 blur2 = cv2.GaussianBlur(image, (5, 5), 0) # 중심에 있는 픽셀에 높은 가중치 blur3 = cv2.medianBlur(image, 5) # 무작위 노이즈 제거에 효과적  cv2.imshow("image", image) cv2.imshow("blur - blur", blur1) cv2.imshow("blur - Gaussian", blur2) cv2.imshow("blur - median", blur3) cv2.waitKey(0) |

14. 캐니에지 알고리즘에서 이중 임계값을 트랙바로 만들어서 두 개의 임계값을 조절하여 에지를 검출하도록 프로그램을 작성하시오.



|  |
| --- |
| import cv2, numpy as np  image = cv2.imread("../images/aircraft.jpg") if image is None:  raise Exception("영상파일 읽기 오류")  blur1 = cv2.blur(image, (3, 3)) # 평균 블러링 blur2 = cv2.GaussianBlur(image, (5, 5), 0) # 중심에 있는 픽셀에 높은 가중치 blur3 = cv2.medianBlur(image, 5) # 무작위 노이즈 제거에 효과적  cv2.imshow("image", image) cv2.imshow("blur - blur", blur1) cv2.imshow("blur - Gaussian", blur2) cv2.imshow("blur - median", blur3) cv2.waitKey(0)  cv2.resizeWindow(title, 400, 300)  while True:  ret, frame = capture.read()  if not ret or cv2.waitKey(delay) >= 0:  break  frame\_cnt += 1  mask = np.zeros(frame.shape[:2], np.uint8)  # 세로열(30~270), 가로행(30~350)  mask[30:30+240, 30:30+320] = 255  frame = cv2.bitwise\_or(frame, frame, mask=mask)  cv2.rectangle(frame, (30, 30), (30+320, 30+240), color=(0, 0, 255), thickness=1)  cv2.imshow(title, frame)  capture.release() |

15. 미디언 필터링을 수행하는 함수를 직접 작성하고, 수행 결과를 윈도우에 표시하시오.



|  |
| --- |
| import numpy as np, cv2  def median\_filter(image, ksize):  rows, cols = image.shape[:2]  dst = np.zeros((rows, cols), np.uint8)  center = ksize // 2   for i in range(center, rows-center):  for j in range(center, cols - center):  y1, y2 = i - center, i + center + 1  x1, x2 = j - center, j + center + 1  mask = image[y1:y2, x1:x2].flatten()   sort\_mask = cv2.sort(mask, cv2.SORT\_EVERY\_COLUMN)  dst[i, j] = sort\_mask[sort\_mask.size//2]  return dst   def salt\_pepper\_noise(img, n):  h, w = img.shape[:2]  x, y = np.random.randint(0, w, n), np.random.randint(0, h, n)  noise = img.copy()  for x, y in zip(x, y):  noise[y, x] = 0 if np.random.rand() < 0.5 else 255  return noise  image = cv2.imread("../images/aircraft.jpg", cv2.IMREAD\_GRAYSCALE) if image is None:  raise Exception("영상파일 읽기 오류")  noise = salt\_pepper\_noise(image, 500) med\_img1 = median\_filter(noise, 5) med\_img2 = cv2.medianBlur(noise, 5)  cv2.imshow("image", image) cv2.imshow("noise", noise) cv2.imshow("median - User", med\_img1) cv2.imshow("median - OpenCV", med\_img2) cv2.waitKey(0) |

16. 예제\_7.4.1과 예제 7.4.2에서 구현한 침식 연산과 팽창 연산 함수 erode() 와 dilate()는 소스 내용이 거의 동일하다. 두 함수를 참고하여 하나로 통일해서 morphology() 함수로 구현하시오

.

|  |
| --- |
| import numpy as np, cv2  def morphology(img, mask, flag = True):  # True : erode, False : dilate  dst = np.zeros(img.shape, np.uint8)  if mask is None:  mask = np.ones((3, 3), np.uint8)  ycenter, xcenter = np.divmod(mask.shape[:2], 2)[0]  mcnt = cv2.countNonZero(mask)  for i in range(ycenter, img.shape[0] - ycenter):  for j in range(xcenter, img.shape[1] - xcenter):  y1, y2 = i - ycenter, i + ycenter + 1  x1, x2 = j - xcenter, j + xcenter + 1  roi = img[y1:y2, x1:x2]  temp = cv2.bitwise\_and(roi, mask)  cnt = cv2.countNonZero(temp)  if flag:  dst[i, j] = 255 if (cnt == mcnt) else 0  else:  dst[i, j] = 0 if (cnt == 0) else 255  return dst   image = cv2.imread("../images/aircraft.jpg", cv2.IMREAD\_GRAYSCALE) if image is None:  raise Exception("영상파일 읽기 오류")  mask = np.array([[0, 1, 0],  [1, 1, 1],  [0, 1, 0]]).astype('uint8') th\_img = cv2.threshold(image, 128, 255, cv2.THRESH\_BINARY)[1] dst1 = morphology(th\_img, mask, False) dst2 = cv2.dilate(th\_img, mask) dst3 = morphology(th\_img, mask, True) dst4 = cv2.erode(th\_img, mask)  cv2.imshow("User dilate", dst1) cv2.imshow("OpenCV dilate", dst2) cv2.imshow("User erode", dst3) cv2.imshow("OpenCV erode", dst4) cv2.waitKey(0) |

17. 심화예제\_7.4.4는 키보드로부터 영상파일의 번호를 입력받아서 열림 연산을 수행한다. 이 예제를 윗쪽과 아래쪽 화살표 키를 이용해서 다음 영상을 로드하여 수행하며, ESC 키를 누르면 종료하도록 수정하시오.

|  |
| --- |
| import numpy as np, cv2 import os path = "../images/plate/" file = os.listdir(path) n = 0 length = len(file)  switch\_case = {  2490368: 1,  2621440: -1 }  while True:  n %= length  fname = file[n]  image = cv2.imread(path + fname, cv2.IMREAD\_COLOR)  if image is None:  raise Exception("영상파일 읽기 실패")  mask = np.ones((5, 17), np.uint8)  gray = cv2.cvtColor(image, cv2.COLOR\_BGR2GRAY)  gray = cv2.blur(gray, (5, 5))  gray = cv2.Sobel(gray, cv2.CV\_8U, 1, 0, 5)  th\_img = cv2.threshold(gray, 120, 255, cv2.THRESH\_BINARY)[1]  morph = cv2.morphologyEx(th\_img, cv2.MORPH\_CLOSE, mask, iterations=3)  image = cv2.hconcat([image,  cv2.cvtColor(th\_img, cv2.COLOR\_GRAY2BGR),  cv2.cvtColor(morph, cv2.COLOR\_GRAY2BGR)])  cv2.imshow("image", image)  key = cv2.waitKeyEx(100)  if key == 27:  break  try:  n += switch\_case[key]  except KeyError:  continue  cv2.destroyAllWindows() |