연습문제 5장

1. OpenCV의 채널 처리 함수에 대해서 아는 대로 기술하시오.

|  |
| --- |
| cv2.merge()  : 여러 개의 채널을 하나의 영상으로 합성  cv2.split()  : 하나의 영상을 여러 개의 채널로 분리 |

2. OpenCV의 사칙 연산을 수행하는 함수와 연산의 수행 방법에 대해서 기술하시오.

|  |
| --- |
| cv2.add()  : 두 개의 배열 혹은 배열과 스칼라의 각 원소간 합을 계산  cv2.subtract()  : 두 개의 배열 혹은 배열과 스칼라의 각 원소 간 차분을 계산.  cv2.multiply()  : 두 배열의 각 원소 간 곱을 계산.  cv2.divide()  : 스칼라값과 행렬원소간 나눗셈을 수행  cv2.addWeighted()  : 두 배열의 각 원소에 가중치를 곱한 후에 각 원소 간 합 즉, 가중된 합을 계산 |

3. 행렬(ndarray)을 초기화하는 방법들에 대해서 기술하고, 각 방법으로 선언하시오.

|  |
| --- |
| (1) np.zeros((2, 4), np.uint8)  : 2\*4 행렬 생성, 0으로 모든 원소를 초기화. type은 uint8  (2) np.ones((2, 4), np.uint8)  : 2\*4 행렬 생성, 1로 모든 원소를 초기화. type은 uint8  (3) np.full((2, 4), 30, np.uint8)  : 2\*4 행렬 생성, 30으로 모든 원소를 초기화. type은 uint8  (4) np.array([1, 2, 3], np.float32)  : 1\*3 행렬 생성, 1., 2., 3.을 원소로 갖는다. type은 float32 |

4. 사칙 연산이나 논리 비트 연산에서 마스킹(masking)을 사용할 수 있다. 마스크에 대한 의미와 사용법에 대해서 설명하시오.

|  |
| --- |
| mask: 연산 마스크 - 0이 아닌 마스크 원소의 위치에서만 연산을 수행(8비트 단일 채널) 하도록 함. cv2.add(src1, src2, mask) 과 같은 형식으로 연산을 하고싶은 위치의 값을 0이 아니도록 변경하고 함수에 전달하면 된다. |

5. cv2.reduce() 함수에 대해서 설명하고, 특히 축소 시에 사용하는 연산 옵션에 대해서 상세히 설명하시오.

|  |
| --- |
| 행렬을 열방향/행방향으로 옵션 상수(rtype)에 따라 축소한다.  rtype : 연산 옵션  (1) cv2.REDUCE\_SUM: 행렬의 모든 행(열)들을 합한다.  (2) cv2.REDUCE\_AVG: 행렬의 모든 행(열)들의 평균을 구한다.  (3) cv2.REDUCE\_MAX: 행렬의 모든 행(열)들의 최댓값을 구한다.  (4) cv2. REDUCE\_MIN: 행렬의 모든 행(열)들의 최솟값을 구한다. |

6. 다음 예시 코드는 컴파일 에러가 발생한다. 에러가 발생하는 부분을 수정하고 실행 결과를 적으시오.

(1)

|  |
| --- |
| import numpy as np, cv2  m1 = np.array([1, 2, 3, 1, 2, 3])  m2 = np.array([3, 3, 4, 2, 2, 3])  m3 = m1 + m2  m4 = m1 - m2  print("[m1] = %s" % m1)  print("[m2] = %s" % m2)  print("[m3] = %s" % m3)  print("[m4] = %s" % m4) |

output:

|  |
| --- |
| [m1] = [1 2 3 1 2 3]  [m2] = [3 3 4 2 2 3]  [m3] = [4 5 7 3 4 6]  [m4] = [-2 -1 -1 -1 0 0] |

(2)

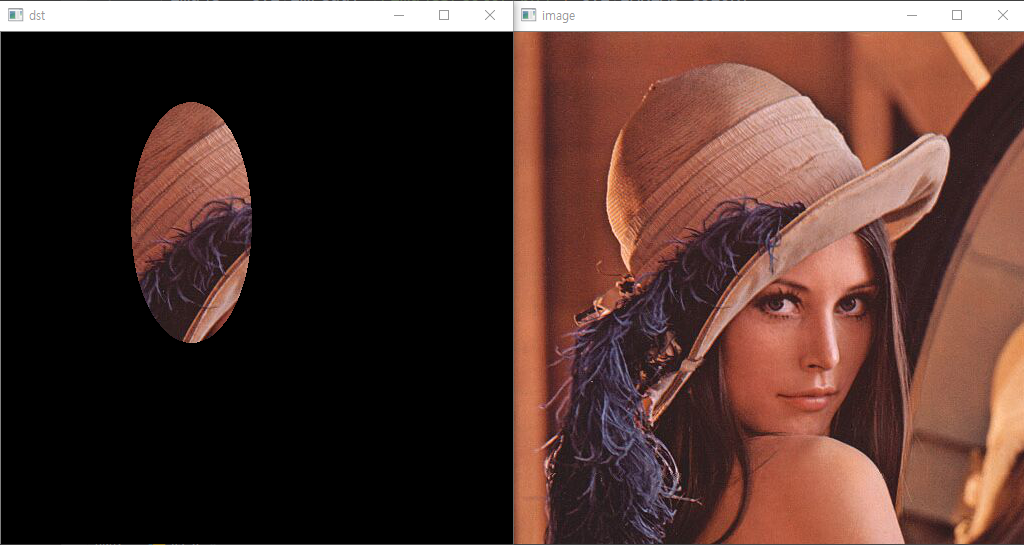
|  |
| --- |
| import numpy as np  import cv2  data = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 2]  m1 = np.array(data).reshape(2, 2, 3)  r, g, b = cv2.split(m1)  print("[m1] = %s" % m1)  print("[r, g, b] = %s, %s, %s" % (r, g, b)) |

|  |
| --- |
| [m1] = [[[ 1 2 3]  [ 4 5 6]]  [[ 7 8 9]  [10 11 2]]]  [r, g, b] = [[ 1 4]  [ 7 10]], [[ 2 5]  [ 8 11]], [[3 6]  [9 2]] |

7. 다음의 컬러 영상파일(logo.jpg)을 입력 받아서 RGB의 3개 채널을 분리하고, 각 채널을 컬러 영상을 윈도우에 표시해보자. 즉, RED 채널은 빨간색으로, Green 채널은 초록색으로, Blue 채널은 파란색으로 표현되도록 다음의 프로그램을 완성하시오.

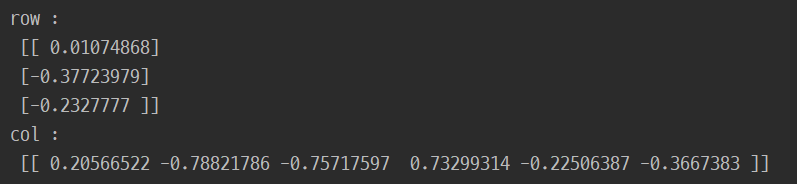
|  |
| --- |
| import cv2  import numpy as np  logo = cv2.imread("./images/logo.jpg")  if logo is None:  raise Exception("영상 읽기 실패")  blue, green, red = cv2.split(logo)  masks = []  for x in [blue, green, red]:  # 임계값 235 이상인 값을 255로 치환. 미만은 0으로.  mask = cv2.threshold(x, 235, 255, cv2.THRESH\_BINARY)[1]  masks.append(mask)  # 마스크 연산 수행 - 마스크 배열의 원소가 0이 아닌 좌표만 계산을 수행  blue\_img = cv2.bitwise\_and(logo, logo, mask=masks[0])  green\_img = cv2.bitwise\_and(logo, logo, mask=masks[1])  red\_img = cv2.bitwise\_and(logo, logo, mask=masks[2])  cv2.imshow('logo', logo)  cv2.imshow("blue\_img", blue\_img)  cv2.imshow("green\_img", green\_img)  cv2.imshow("red\_img", red\_img)  cv2.waitKey(0)  cv2.destroyAllWindows() |

8. 다음 영역에서 특정 영역의 타원만을 복사하여 새 창에 표시하는 프로그램을 완성하시오.  
힌트 >> 마스크 행렬을 이용한다.  
- cv2.bitwise\_and(), cv2.bitwise\_or() 등의 함수를 사용해서 같은 결과(dst)가 나도록 구성한다.  
- 타원 영역의 중심과 크기는 임의로 지정한다.



|  |
| --- |
| import numpy as np  import cv2  image = cv2.imread("./images/color.png", cv2.IMREAD\_COLOR)  if image is None:  raise Exception("영상파일 읽기 오류")  mask = np.zeros(image.shape[:2], np.uint8)  cv2.ellipse(mask, (190, 190), (60, 120), 0, 0, 360, (255, 255, 255), -1)  dst = cv2.bitwise\_or(image, image, mask=mask)  cv2.imshow("image", image)  cv2.imshow("dst", dst)  cv2.waitKey(0)  cv2.destroyAllWindows() |

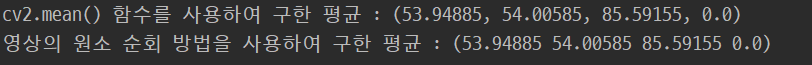
9. 3행, 6열의 행렬을 생성하고, 행렬의 원소를 초기화한 후에 cv2.reduce() 함수를 이용해서 가로 방향과 세로 방향으로 감축하여 평균을 구한 결과를 출력하시오.



|  |
| --- |
| import numpy as np  import cv2  x = np.random.normal(0, 1, (3, 6))  # 행 단위(가로 방향)으로 감축  row = cv2.reduce(x, dim=1, rtype=cv2.REDUCE\_AVG)  # 열 단위(세로 방향)으로 감축  col = cv2.reduce(x, dim=0, rtype=cv2.REDUCE\_AVG)  print("row : \n {}".format(row))  print("col : \n {}".format(col)) |

10. PC 카메라로 영상을 읽어서 특정 부분(관심 영역)의 합과 평균을 구하는 프로그램을 작성하시오.

1) 관심 영역은 200, 100 좌표에서 200x100 크기로 한다.  
2) cv2.mean() 함수를 사용하여 평균을 구하시오.  
3) cv2.mean() 함수를 사용하지 않고 영상의 원소 순회 방법으로 평균을 구하시오.



|  |
| --- |
| import numpy as np  import cv2  image = cv2.imread("./images/color.png", cv2.IMREAD\_COLOR)  if image is None:  raise Exception("영상 읽기 에러")  mask = np.zeros(image.shape[:2], np.uint8)  mask[200:400, 100:200] = 255  mean\_value = cv2.mean(image, mask)  r, g, b = list(map(np.mean, cv2.split(image[200:400, 100:200])))  print("cv2.mean() 함수를 사용하여 구한 평균 : {}".format(mean\_value))  print("영상의 원소 순회 방법을 사용하여 구한 평균 : ({} {} {} 0.0)".format(r, g, b)) |

11. PC 카메라로 영상을 받아들여서 다음과 같이 윈도우의 특정 영역에서 재생하시오.

1) 메인 윈도우는 400x300 크기로 한다.  
2) 관심 영역은 30, 30 좌표에서 320x240 크기로 한다.  
3) 관심 영역에 빨간색 테두리를 두른다.

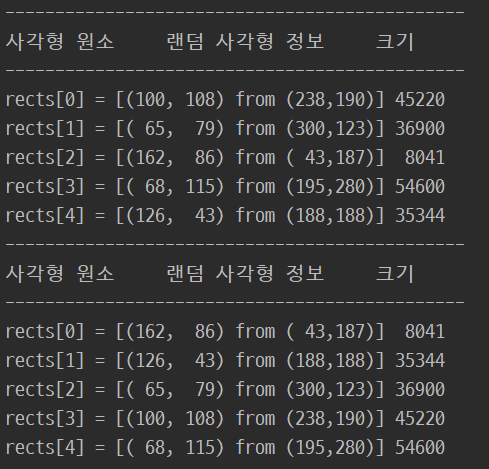
|  |
| --- |
| import cv2  import numpy as np  capture = cv2.VideoCapture("images/video\_file.avi")  if not capture.isOpened():  raise Exception("동영상 파일 개방 안됨")  title = "Main Window"  frame\_rate = capture.get(cv2.CAP\_PROP\_FPS)  delay = int(1000 / frame\_rate)  frame\_cnt = 0 # 현재 프레임 번호  cv2.namedWindow(title, cv2.WINDOW\_NORMAL)  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() |

12. 영상파일을 읽어서 메인 윈도우에 다음과 같이 출력하시오.

1) 메인 윈도우의 특정 부분 2곳을 관심 영역으로 지정한다.  
2) 관심 영역1는 영상의 밝기를 50만큼 밝게 한다.  
3) 관심 영역2는 영상의 화소대비를 증가시킨다. 

|  |
| --- |
| import cv2  import numpy as np  capture = cv2.VideoCapture("images/video\_file.mp4")  if not capture.isOpened():  raise Exception("동영상 파일 개방 안됨")  title = "Main Window"  frame\_rate = capture.get(cv2.CAP\_PROP\_FPS)  delay = int(1000 / frame\_rate)  frame\_cnt = 0 # 현재 프레임 번호  cv2.namedWindow(title, cv2.WINDOW\_NORMAL)  cv2.resizeWindow(title, 960, 540)  while True:  ret, frame = capture.read()  if not ret or cv2.waitKey(delay) >= 0:  break  frame\_cnt += 1  mask = np.zeros(frame.shape, np.uint8)  mask[100:200, 100:100+200] = 50  frame = cv2.add(frame, mask)  frame = cv2.rectangle(frame, (100, 100), (300, 200), (0, 0, 255), thickness=5)  tmp = cv2.multiply(frame, 1.5)  frame[500:700, 500:600] = tmp[500:700, 500:600]  frame = cv2.rectangle(frame, (500, 500), (600, 700), (0, 0, 255), thickness=5)  cv2.imshow(title, frame)  capture.release() |

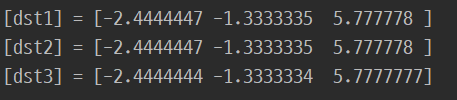
13. cv2.sortIdx() 함수를 활용해서 다음의 조건에 부합하도록 벡터의 원소를 정렬하시오.

1) 벡터의 원소는 Rect 객체이다.  
2) 벡터의 원소는 임의로 지정한다.  
3) 정렬의 기준은 Rect 객체의 크기이다.  
4) 오름차순으로 정렬하여 콘솔창에 출력한다.  
  
15.sortidx\_rect.py와 같아 보이는데.. 아닌가?  


|  |
| --- |
| import numpy as np, cv2  def print\_rects(rects):  print("-" \* 46)  print("사각형 원소\t\t랜덤 사각형 정보\t 크기")  print("-" \* 46)  for i, (x, y, w, h, a) in enumerate(rects):  print("rects[%i] = [(%3d, %3d) from (%3d,%3d)] %5d" % (i, x, y, w, h, a))  rands = np.zeros((5, 5), np.uint16)  starts = cv2.randn(rands[:, :2], 100, 50)  ends = cv2.randn(rands[:, 2:-1], 300, 50)  sizes = cv2.absdiff(starts, ends)  areas = sizes[:, 0] \* sizes[:, 1]  rects = rands.copy()  rects[:, 2:-1] = sizes  rects[:, -1] = areas  idx = cv2.sortIdx(areas, cv2.SORT\_EVERY\_COLUMN).flatten()  print\_rects(rects)  print\_rects(rects[idx.astype('int')]) |

14. 다음의 연립방정식을 가우시안 소거법의 역함수를 계산해서 해를 구하는 프로그램을 작성하시오.

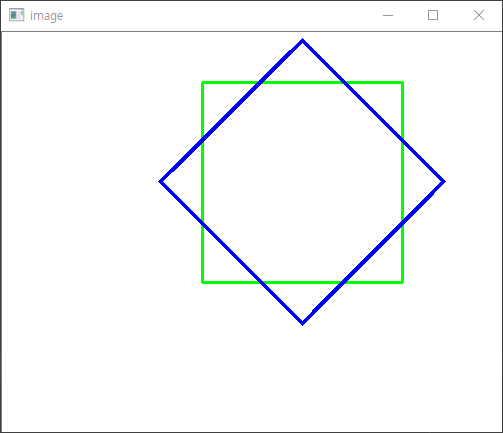
연립방정식:  
3x1+6x2+3x3=2  
-5x1+6x2+x3=10  
2x1-3x2+5x3=28



|  |
| --- |
| import numpy as np  import cv2  data = [3, 6, 3, -5, 6, 1, 2, -3, 5]  m1 = np.array(data, np.float32).reshape(3, 3)  m2 = np.array([2, 10, 28], np.float32)  ret, inv = cv2.invert(m1, cv2.DECOMP\_LU)  if ret:  dst1 = inv.dot(m2)  dst2 = cv2.gemm(inv, m2, 1, None, 1)  \_, dst3 = cv2.solve(m1, m2, cv2.DECOMP\_LU)  print("[dst1] =", dst1.flatten())  print("[dst2] =", dst2.flatten())  print("[dst3] =", dst3.flatten())  else:  print("역행렬이 존재하지 않습니다.") |

15. 5.6절에서 예제\_5.6.2의 사각형 회전하기 예제를 확장하여 그 사각형의 중심점을 기준으로 45도 회전시키는 프로그램을 완성하시오.

힌트 >> 원점을 변환할 것이 아니라 사각형의 중심점을 원점으로 만들면 된다. 사각형의 중심점을 원점으로 만드는 것은 좌표의 평행이동이다.  
1) 예제\_5.6.2는 변환 행렬이 2x2이지만, 본 문제는 변환 행렬이 3x3 행렬  
2) 원점으로 평행이동 -> 회전변환 -> 역 평행이동  
3. 세 번의 변환은 각 변환 행렬의 곱으로 나타낼 수 있음  
 전체 변환 행렬 = 이동 변환 행렬 \* 회전 변환 행렬 \* 이동 변환 행렬



|  |
| --- |
| import numpy as np  import cv2  pts1 = np.array([(200, 50, 1), (400, 50, 1),  (400, 250, 1), (200, 250, 1)], np.float32)  theta = 45 \* np.pi / 180  m = np.array([[np.cos(theta), -np.sin(theta), 0],  [np.sin(theta), np.cos(theta), 0],  [0, 0, 1]], np.float32)  delta = (pts1[2] - pts1[0])//2  center = pts1[0] + delta  t1 = np.eye(3, dtype=np.float32)  t2 = np.eye(3, dtype=np.float32)  t1[0:2, 2] = np.mean(pts1, axis=0)[:-1]  t2[0:2, 2] = -np.mean(pts1, axis=0)[:-1]  m2 = np.dot(t1, m)  m2 = np.dot(m2, t2)  pts2 = cv2.gemm(pts1, m2, 1, None, 1, flags=cv2.GEMM\_2\_T)  for i, (pt1, pt2) in enumerate(zip(pts1, pts2)):  print("pts1[%d] = %s, pts2[%d]= %s" % (i, pt1, i, pt2))  image = np.full((400, 500, 3), 255, np.uint8)  cv2.polylines(image, [np.int32(pts1[:, :2])], True, (0, 255, 0), 2)  cv2.polylines(image, [np.int32(pts2[:, :2])], True, (255, 0, 0), 3)  cv2.imshow("image", image)  cv2.waitKey(0)  cv2.destroyAllWindows() |