연습문제 8장

1. 순방향 사상과 역방향 사상에 대해 설명하고, 장단점을 비교하시오.

|  |
| --- |
| 순방향 사상은 입력 영상의 좌표를 중심으로 목적 영상의 좌표를 계산하여 화소의 위치를 변환하는 방식이다. 입력 영상과 목적 영상이 크기가 같을 때 유용하다. 그러나 영상을 확대하거나 회전할 때 홀이 발생하거나, 영상을 축소할 때 오버랩이 발생할 수 있다.  역방향 사상은 목적 영상의 좌표를 중심으로 역 변환을 계산하여 해당하는 원본 영상의 좌표를 찾아서 화소 값을 가져오는 방식이다. 홀이나 오버랩은 발생하지 않지만, 입력 영상의 한 화소를 목적 영상의 여러 화소에서 사용하게 된다면 결과 영상의 품질이 떨어질 수 있다. |

2. 홀(hole)과 오버랩(overlap)에 대해 설명하시오.

|  |
| --- |
| 홀은 입력 영상의 좌표들로 목적 영상의 좌표를 만드는 과정에서 사상되지 않은 화소를 가리킨다. 보통 영상을 확대하거나 회전할 때 발생한다.  오버랩은 입력 영상의 여러 화소들이 목적 영상의 한 화소로 사상되는 것을 말한다. 영상을 축소할 때 주로 발생한다. |

3. 보간법이 필요한 이유를 설명하고, OpenCV에서 보간 방법을 가리키는 옵션 상수를 설명하시오.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 목적 영상에서 홀의 화소들을 채우며, 오버랩 되지 않게 화소들을 배치하여 목적 영상을 만들기 위해 보간법을 사용한다.  보간 방법 옵션 상수   |  |  |  | | --- | --- | --- | | 옵션 상수 | 값 | 설명 | | cv2.INTER\_NEAREST | 0 | 최근접 이웃 보간 | | cv2.INTER\_LINEAR | 1 | 양선형 보간 (기본값) | | cv2.INTER\_CUBIC | 2 | 바이큐빅 보간 – 4 x 4 이웃화소 이용 | | cv2.INTER\_AREA | 3 | 픽셀 영역의 관계로 리샘플링 | | cv2.INTER\_LANCZOS4 | 4 | Lanczos 보간 – 8 x 8 이웃 화소 이용 | |

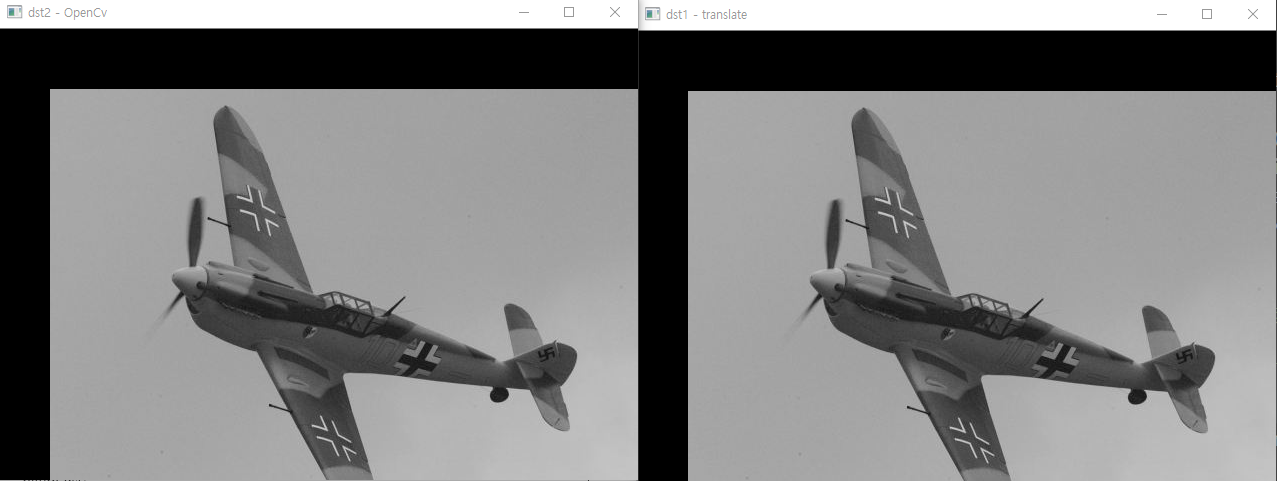
4. 최근접 이웃 보간법에 대해 기술하시오.

|  |
| --- |
| 최근접 이웃 보간법은 목적 영상을 만드는 과정에서 홀이 되어 할당받지 못한 화소들의 값을 찾을 때, 목적 영상의 화소에 가장 가깝게 이웃한 입력 영상의 화소값을 가져오는 방법이다. |

5. 양선형 보간법의 과정을 상세히 설명하시오.

|  |
| --- |
| 양선형 보간법은 선행 보간을 두 번에 걸쳐서 수행하는 방법이다.  (1) 먼저, 목적 영상의 화소를 역변환으로 계산하여 가장 가까운 위치에 있는 입력 영상의 4개 화소를 가져온다.  (2) 4개 화소를 두 개씩 묶어서 화소값으로 두화소를 잇는 직성을 구성한다.  (3) 직선의 선상에서 목적 영상 화소의 좌표로 중간 위치를 찾고, 그 위치의 화소값을 계산한다. 이 때, 거리 비율을 바탕으로 직선의 수식을 이용해서 계산한다.  (4) 마지막으로 구해진 중간화소값을 잇는 직선을 다시 구성하고, 두개의 중간 화소값과 거리 비율을 바탕으로 직선의 수식을 이용해서 최종 화소값을 계산한다. 이 최종 화소값이 목적 영상의 해당 좌표의 화소값이 된다. |

10. 원본 영상에 (50, 60) 좌표만큼 평행이동을 수행하는 프로그램을 작성하시오. 직접 translate() 함수를 작성한 결과와 OpenCV 함수를 사용한 결과를 모두 표시하시오.



|  |
| --- |
| import numpy as np, cv2  def contain(p, shape):  return 0 <= p[0] < shape[0] and 0 <= p[1] < shape[1]  def translate(img, pt):  dst = np.zeros(img.shape, img.dtype)  for i in range(img.shape[0]):  for j in range(img.shape[1]):  x, y = np.subtract((j, i), pt)  if contain((y, x), img.shape):  dst[i, j] = img[y, x]  return dst  image = cv2.imread("../images/aircraft.jpg", cv2.IMREAD\_GRAYSCALE) if image is None:  raise Exception("영상파일 읽기 에러")  dst1 = translate(image, (50, 60)) M = np.float32([[1, 0, 50], [0, 1, 60]]) dst2 = cv2.warpAffine(image, M, image.shape[::-1]) cv2.imshow("dst1 - translate", dst1) cv2.imshow("dst2 - OpenCv", dst2) cv2.waitKey(0) |