Skeleton 알고리즘을 이용하여 Skeleton 구하기

(시각개론 과제2)

201124506 이한결

# 프로젝트 목적

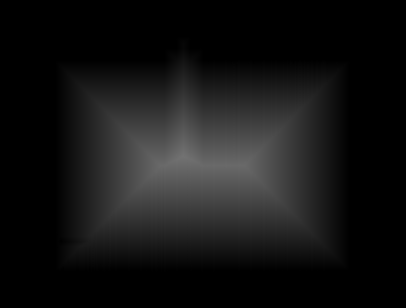
* 본 프로젝트는 Skeleton 알고리즘을 이용하여 주어진 그림에서 Skeleton을 찾아내는 것이다.

# 시스템환경

|  |
| --- |
| Windows 8  Python 2.7  openCV 2.4.13  numpy 1.11.1 |

# 구현 결과

## 3-1. Distance Transform



『그림 1』

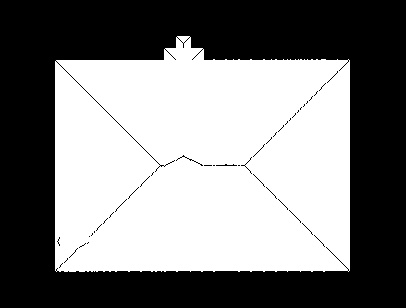


『그림 2』

|  |
| --- |
| img = cv2.imread("skeleton.jpg",0)  \_,thresh = cv2.threshold(img,150,255,cv2.THRESH\_BINARY)  rows,cols = thresh.shape  for i in range(rows):  for j in range(cols):  if thresh[i,j] == 0:  thresh[i,j] = 1  else:  thresh[i,j] = 0  def distanceTransform(img):  rows,cols = img.shape  ret = copy.deepcopy(img)  it = cols/2  for \_ in range(it):  for i in range(1,rows-1):  for j in range(1,cols-1):  tmp = min(ret[i,j],ret[i-1,j],ret[i+1,j],ret[i,j-1],ret[i,j+1])  if ret[i,j] == tmp:  ret[i,j] = img[i,j] + tmp  return ret |

그림1은 직접 구현하여 출력한 것이고 그림2는 opencv에 있는 distanceTransform함수를 이용하여 출력한 것이다. (cv2.distanceTransform(thresh,cv2.cv.CV\_DIST\_L2,5)

## 3-2. Skeleton



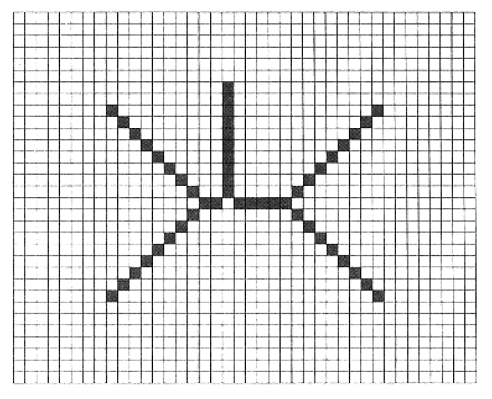
『그림 3』

|  |
| --- |
| def getSkeleton(img):  rows,cols = img.shape  ret = copy.deepcopy(img)  for i in range(1,rows-1):  for j in range(1,cols-1):  tmp = max(img[i][j],img[i-1,j],img[i+1,j],img[i,j-1],img[i,j+1])  if img[i,j] == tmp:  ret[i,j] = 0  else:  ret[i,j] = 255  return ret |

# 결론

본 프로젝트를 통하여 DistanceTransform 과 Skeleton 알고리즘에 관해 깊게 이해할 수 있게 되었다. DistanceTransform을 구현할 때 기본 이미지와 연산을 한다는 것을 자꾸 헷갈려서 어려움이 있었지만 opencv에 존재하는 distanceTransform함수와 결과를 비교해 가면서 수정해나가니 마무리가 잘 되었다.

Skeleton알고리즘을 이용하여 주어진 사진의 Skeleton을 구하니 예상했던 결과인 그림4와는 전혀 다른 그림이 나왔다.



『그림 4』