In [1]:

```
import cv2
import numpy as np
import matplotlib.pyplot as plt
import sklearn.metrics as metrics
```

In [2]:

```
def calcSim(target, predict):

    if len(target) != len(predict):
        return -1

    num_target = 0
    num_predict = 0
    num_union = 0
    for i in range(len(target)):
        if (target[i] != 0):
            num_target = num_target + 1
        if (predict[i] != 0):
            num_predict = num_predict + 1
        if (target[i] != 0) and (target[i] == predict[i]):
            num_union = num_union + 1

    sim = 2*(num_union/(num_target+num_predict))

    return sim
```

In [3]:

```
src = cv2.imread('./data/cardiac_cta.bmp', cv2.IMREAD_GRAYSCALE)
target = cv2.imread('./data/cardiac_label.bmp', cv2.IMREAD_GRAYSCALE)
```

In [4]:

```
## TODO : Segmentation
ret2, thresh_mask = cv2.threshold(src, 180, 255, cv2.THRESH_BINARY)
print('ret2=', ret2)
##
```

```
ret2= 180.0
```

In [5]:

```
kernel= cv2.getStructuringElement(shape=cv2.MORPH_RECT, ksize=(3,3))
erode = cv2.erode(thresh_mask, kernel,iterations = 3)
dilate = cv2.dilate(erode,kernel,iterations = 3)
cv2.imshow('src',    src)
cv2.imshow('dilate', dilate)
cv2.waitKey()
cv2.destroyAllWindows()
```

In [8]:

```
ret, labels, stats, centroids = cv2.connectedComponentsWithStats(dilate)
print('ret =', ret)
print('stats.shape =', stats.shape)
print('stats =', stats)
print('centroids =', centroids)

maxval = 0
maxlabel = 0
for i in range(1, ret):
    if maxval < stats[i][4]:
        maxval = stats[i][4]
        maxlabel = i

dst = np.zeros(dilate.shape, dtype=dilate.dtype)
for i in range(1, ret): # 분할영역 표시
    dst[labels == maxlabel] = 255

predict = dst
```

```
ret = 8
stats.shape = (8, 5)
stats = [[    0     0   256   256 58240]
 [   97    43    18    21   309]
 [   93    67     9     8    70]
 [  119    69    16    13   179]
 [   71    80   118    96  5753]
 [  113   182    27    28   618]
 [   67   233    13    20   216]
 [  106   236    12    15   151]]
centroids = [[127.2848386  126.05036058]
 [106.14239482  53.36893204]
 [ 97.1         70.6       ]
 [126.30726257  74.70391061]
 [133.85277247 133.88162698]
 [126.09708738 195.17799353]
 [ 73.03703704 242.28703704]
 [111.31125828 242.9602649 ]]
```

In [9]:

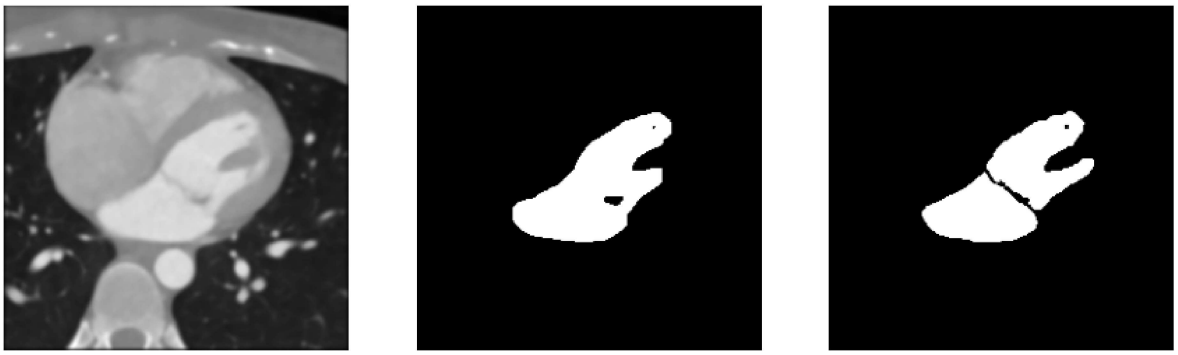
```
plt.figure(figsize=(15, 15))
plt.subplot(1, 3, 1)
plt.axis('off')
plt.imshow(src, cmap='gray')

plt.subplot(1, 3, 2)
plt.axis('off')
plt.imshow(predict, cmap='gray')

plt.subplot(1, 3, 3)
plt.axis('off')
plt.imshow(target, cmap='gray')
```

Out[9]:

```
<matplotlib.image.AxesImage at 0x16508ab6bb0>
```



In [10]:

```
target = target.flatten()
predict = predict.flatten()
print(target.shape)
print(predict.shape)

target[target[:] == 255] = 1
predict[predict[:] == 255] = 1

sim = calcSim(target, predict)
print("similarity: ", sim)
```

```
(65536,)
(65536,)
similarity:  0.9529992209815632
```

In [ ]: