HW-13

```
import cv2
import sys
import copy
import numpy as np
import itertools, math
import time
import matplotlib.pyplot as plt

def get_cluster(originalImage, num_clusters):

    Z = originalImage.reshape((-1, 3))
    Z = np.float32(Z)
    criteria = (cv2.TERM_CRITERIA_EPS + cv2.TERM_CRITERIA_MAX_ITER, 10, 1.0)
    ret, label, center = cv2.kmeans(Z, num_clusters, None, criteria, 10,
cv2.KMEANS_RANDOM_CENTERS)

    # Now convert back into uint8, and make original image
    center = np.uint8(center)
    res = center[label.flatten()]
    res2 = res.reshape((originalImage.shape))

    # cv2.imshow('res2',res2)
    # cv2.waitKey(0)
```

```
    # cv2.destroyAllWindows()

    return center, label

def norm(a, b):
    return sum([(i - j)**2 for i, j in zip(a, b)]) ** 0.5

def contrast_cue(num_clusters, label, center):
    center = np.int32(center)
    N = label.shape[0]
    contrast_cue_list = []
    frequency_pixels = [0] * (num_clusters)

    for i in label:
       frequency_pixels[i[0]] += 1

    for k in range(num_clusters):
       cue_k = 0
       for i in range(num_clusters):
             if i != k:
                   cue_k += frequency_pixels[i] * norm(center[k], center[i])

       contrast_cue_list.append(cue_k / N)

    return contrast_cue_list

def spatial_cue(num_clusters, label, center, originalImage):
    height, width = originalImage.shape[:2]
    spatial_cue_list = []
    pixel_label = [[] for _ in range(num_clusters)]
    sigma = 100
    frequency_pixels = [0] * (num_clusters)
    center_of_image = [height // 2, width // 2]

    for i in label:
       frequency_pixels[i[0]] += 1

    c = 0
    for i in range(height):
       for j in range(width):
             pixel_label[label[c][0]].append((i, j))
             c += 1
```

```
    for k in range(num_clusters):
       cue = 0
       for pixel_loc in pixel_label[k]:
             cue += math.exp(-(norm(center_of_image ,pixel_loc)/sigma))
       spatial_cue_list.append(cue / frequency_pixels[k])

    return spatial_cue_list



def main(image_path):
    originalImage = cv2.imread(image_path)
    num_clusters = 20
    center, label = get_cluster(originalImage, num_clusters)
    contrast_cue_list = contrast_cue(num_clusters, label, center)
    spatial_cue_list = spatial_cue(num_clusters, label, center, originalImage)

    combined_cue_list = [i*j for i, j in zip(contrast_cue_list, spatial_cue_list)]
    max_cue = max(combined_cue_list)
    for i in range(num_clusters):
       combined_cue_list[i] = int((combined_cue_list[i] / max_cue) * 255)

    height, width = originalImage.shape[:2]

    c = 0
    for i in range(height):
       for j in range(width):
             originalImage[i][j] = combined_cue_list[label[c][0]]
             c += 1

    cv2.imshow('saliency',originalImage)
    cv2.waitKey(0)
    cv2.destroyAllWindows()


if __name__ == '__main__':
    if len(sys.argv) != 2:
       print("Usage: python3 code.py <path_of_image>")
       exit(1)

    image_path = sys.argv[1]
    main(image_path)
```