AML HW8 EM Algorithm

Yu Che Wang / yuchecw2

1 Tree segmented images

Figure 1: Tree image with 10 segments

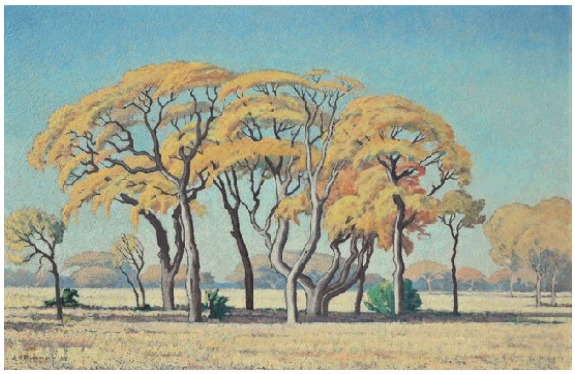

Figure 2: Tree image with 20 segments

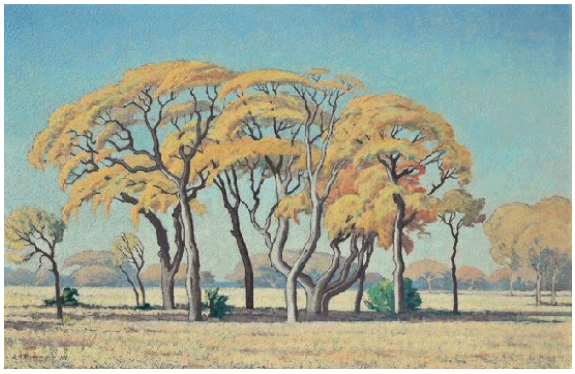

Figure 3: Tree image with 50 segments

2 RobertMixed segmented images

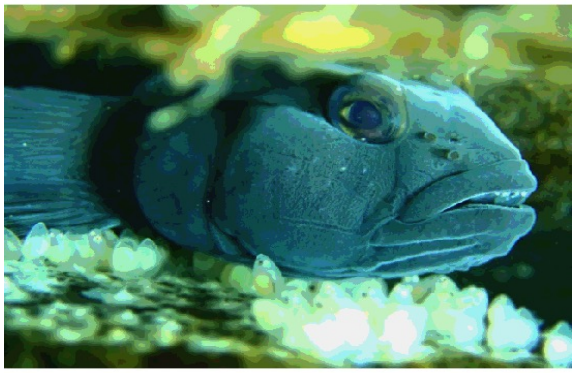

Figure 4: RobertMixed image with 10 segments

Figure 5: RobertMixed image with 20 segments

Figure 6: RobertMixed image with 50 segments

3 Strelitzia segmented images

Figure 7: Strelitzia image with 10 segments

Figure 8: Strelitzia image with 20 segments

Figure 9: Strelitzia image with 50 segments

4 Sunset segmented images

Figure 10: Sunset image with 10 segments

Figure 11: Sunset image with 20 segments

Figure 12: Sunset image with 50 segments

5 Tree segmented images with different starting points

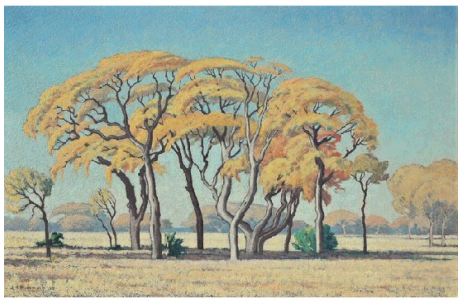

Figure 13: Tree-1

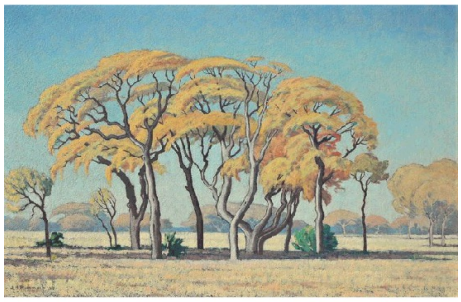

Figure 14: Tree-2

Figure 15: Tree-3

Figure 16: Tree-4

Figure 17: Tree-5

6 Code snippets for EM initialization and updates

```
# Initialization
orig_img, unscaled_img, img = load_image(filename)
nrows, ncols, _ = orig_img.shape
kmeans = KMeans(n_clusters=n_segments, random_state=random_seed).fit(img)
blob_weight = np.zeros(n_segments)   # pi, i.e. weight of each blob
counter = Counter(kmeans.labels_)
for i in range(n_segments):
    blob_weight[i] = counter[i] / img.shape[0]
blob_center = kmeans.cluster_centers_ # mu, i.e. mean of each normal distribution
w = np.zeros((img.shape[0], n_segments))
Q_prev = 0
num_epochs = 30
epsilon = 0.1
# EM Algorithm
for epoch in range(num_epochs):
    # E step
    for i in range(img.shape[0]):
        total = 0
        for j in range(n_segments):
            w[i,j] = np.exp(-0.5*(np.dot(img[i,:]-blob_center[j,:], img[i,:]-blob_center[j,:])))*blob_weight[j]
            total += w[i,j]
        w[i,:] = w[i,:] / total
    # M step
    for j in range(n_segments):
        total = np.zeros((1,3))
        for i in range(img.shape[0]):
            total += img[i,]*w[i,j]
        total = total/(np.sum(w[:,j]))
        blob_center[j] = total
        blob_weight[j] = np.sum(w[:,j]) / img.shape[0]
    # Calculate Q
    Q = 0
    for i in range(img.shape[0]):
        for j in range(n_segments):
            Q += (-0.5*(np.dot(img[i,:]-blob_center[j,:], img[i,:]-blob_center[j,:])) + np.log(blob_weight[j]))*w[i,j]
    # Check if converge or not
    if np.abs(Q - Q_prev) < epsilon:
        break
    Q_prev = Q
```

Figure 18: Code snippets

7 Other relevant code (optional)

```python
def load_image(filename):
    img = Image.open(filename)
    img_arr = np.asarray(img, dtype='int32')
    img_ravel = np.concatenate((img_arr[:,:,0].ravel().reshape(-1,1), img_arr[:,:,1].ravel().reshape(-1,1), img_arr[:,:,2].ravel().reshape(-1,1)), axis=1)
    return img_arr, img_ravel, img_scaled # (H*W, C)
```

Figure 19: Load image to numpy array

```python
cluster_index = [np.argsort(w[i,:])[-1] for i in range(img.shape[0])]
segmented_img_ravel = np.zeros(img.shape)
for i in range(img.shape[0]):
    segmented_img_ravel[i,:] = blob_center[cluster_index[i],:]
segmented_img = np.zeros(orig_img.shape)
for i in range(img.shape[0]):
    segmented_img[i//ncols, i%ncols, :] = segmented_img_ravel[i,:]
segmented_img = Image.fromarray(segmented_img.astype(np.uint8),'RGB')
segmented_img.save('{}_segment-{}_seed-{}.jpg'.format(image_name, n_segments, random_seed))
```

Figure 20: Obtain segment image

```python
image_name = ['RobertMixed03','smallsunset','smallstrelitzia', 'tree']
filename = [i+'.jpg' for i in image_name]
n_segments_list = [10,20,50]

for i in range(4):
    for n_segments in n_segments_list:
        image_segmentation(image_name[i], filename[i], n_segments, 0)
        print('Image: {}, num_segments: {}'.format(image_name[i], n_segments))

for random_seed in [1,2,3,4,5]:
    image_segmentation(image_name[3], filename[3], 20, random_seed)
    print('Tree image with random seed: {}'.format(random_seed))
```

Figure 21: Main method