Computer Vision
Exercise 8

Xingze Tian 17-941-527

December 21, 2017

1 Image Preprocessing

1.1 Gaussian Filter

This is implemented in `gaussianFilter.m` where the method `fspecial()` is used to define the window size and σ. The `imfilter()` function is then used to apply the filter.

1.2 Convert to L*a*b space

This is implemented in `conver2lab.m` where the functions `makecform()` and `applycform()` are used. The L*a*b* color space is used as it is perceptually uniform, that movement in any direction results in an equally perceptible color shift.

2 Mean-Shift Segmentation

2.1 `find_peak`

This is implemented in `find_peak.m`.

For the input X, x_l and r, the function computes the distances between the pixel x_l and all the other pixels in X. Pixels within radius r are the neighbors of the pixel and the mean inside the spherical window is calculated. If the distance between the mean and the pixel is within a threshold, the peak is then the mean. Otherwise, shift the window to the mean and repeat until convergence.

The threshold value chosen in this exercise is 0.001.

2.2 `mean_shift`

This is implemented in `mean_shift.m`.

For each pixel, the `find_peak()` method is used to find . A boolean vector is used (size $L \times 1$) to record if a peak has already been merged. For each peak, if it has not been merged, it is merged with the other peaks that are within a distance less than r/2.

2.3 `meanshiftSeg`

This is implemented in `meanshiftSeg.m`.

This function takes in an L*a*b* image as input. It first uses the `reshape()` function to convert the image to the probability density function of size $L \times n$, where L is the total number of all the pixels and $n = 3$. Then the `mean_shift()` function is used to find the *map* and *peaks* value, where *map* stores the id of the associated peak for each pixel and *peaks* contains the merged peaks. Note *map* is reshaped to the image size afterwards.

2.4 Evaluation

In this exercise, r is chosen to be 7. The segmentation results are shown as below:

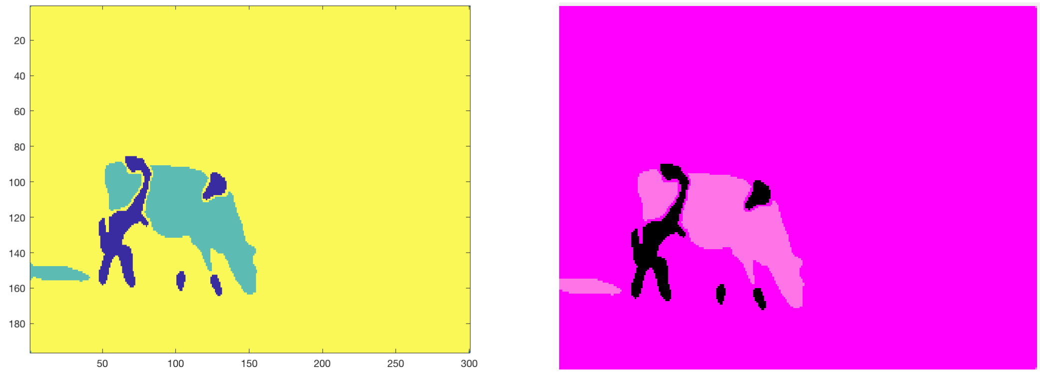


Figure 1: Segmentation results

It is clear from the diagrams that the image is classified to 3 components: the grass field, the black part of the cow and the white part of the cow. Adjusting the size of r gives different results. Increasing r identifies fewer components, as more peaks are merged together. When r is small, fewer peaks will be merged together, thus the total number of peaks is higher.

3 EM Segmentation

3.1 Expectation

This is implemented in `expectation.m`.

The probability is computed as:

$$\gamma_{lk} = P(z_l - k|\mathbf{X}, \mu, \Sigma, \alpha) = \frac{\alpha_k N(x_l|\mu_k, \Sigma_k)}{\sum_{k=1}^{K} \alpha_k N(x_l|\mu_k, \Sigma_k})$$

where

$$N(\mathbf{x}|\mu_k, \Sigma_k) = \frac{1}{(2\pi)^{n/2} det(\Sigma)^{1/2}} exp(-\frac{1}{2}(\mathbf{x} - \mu_k)^T \Sigma_k^{-1} (\mathbf{x} - \mu_k))$$

3.2 Maximization

This is implemented in `maximization.m`.

Th model parameters are updated as:

$$\alpha_k = \frac{1}{L} \sum_{l=1}^{L} \gamma_{lk}, \quad \mu_k = \frac{\sum_{l=1}^{L} x_l \gamma_{lk}}{\sum_{l=1}^{L} \gamma_{lk}}, \quad \Sigma_k = \frac{\sum_{l=1}^{L} \gamma_{lk} (x_k - \mu_k)(x_l - \mu_k)^T}{\sum_{l=1}^{L} \gamma_{lk}}$$

3.3 Segmentation Algorithm

This is implemented in `EM.m`.

The input image is first reshaped as in section 2.3. The parameters are initialized in `generate_mu()` and `generate_cov()`:

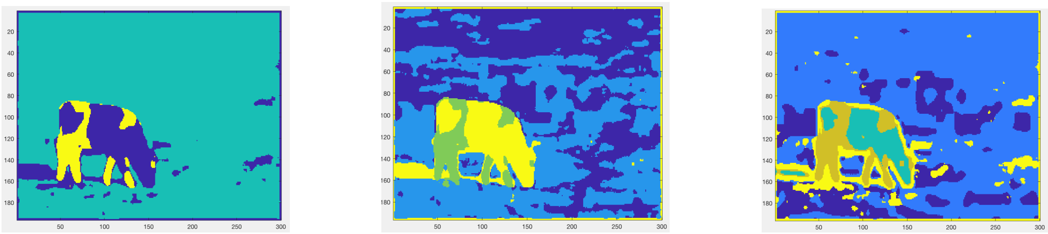
$$\alpha_k = \frac{1}{K}, \quad \Sigma_k = \begin{bmatrix} L* & 0 & 0 \\ 0 & a* & 0 \\ 0 & 0 & b* \end{bmatrix}, \quad \mu_k = \begin{bmatrix} random\_x \\ random\_y \\ random\_z \end{bmatrix}$$

The random values used in μ_k are generated uniformly in the L*a*b* space.

In the iteration step, the `expection()` and `maximization()` functions are used. The function terminates when the μ value doesn't change within a threshold (here, it is set to be 0.001). The result *cluster* equals μ and *map* stores the index where the biggest probability is achieved. Note *map* is reshaped to the image size afterwards.

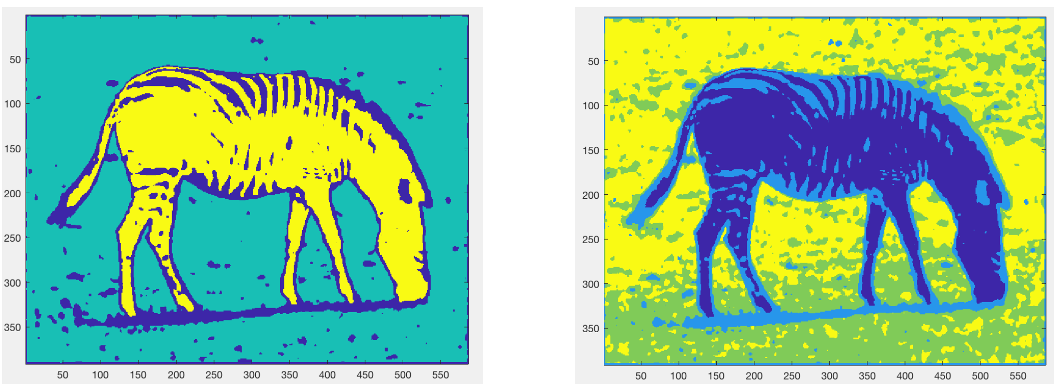
3.4 Evaluation

The segmentation of the results are as shown below:



(a) K = 3 (b) K = 4 (c) K = 5

Figure 2: Segmentation results for the cow image



(a) K = 3 (b) K = 4

Figure 3: Segmentation results for the zebra image

The parameters used for the cow image are as the following:

K = 3

$$\mu = \begin{bmatrix} 19.1812 & -6.6358 & 11.5363 \\ 35.0596 & -13.5923 & 21.0722 \\ 52.3246 & -3.1800 & 12.9429 \end{bmatrix} \quad \alpha = \begin{bmatrix} 0.1511 \\ 0.8082 \\ 0.0408 \end{bmatrix} \quad cov(:,:,1) = \begin{bmatrix} 112.2566 & -54.4466 & 85.2860 \\ -54.4466 & 35.4669 & -47.8578 \\ 85.2860 & -47.8578 & 72.9312 \end{bmatrix}$$

$$cov(:,:,2) = \begin{bmatrix} 8.1590 & 0.0349 & 0.1366 \\ 0.0349 & 0.7310 & -0.1441 \\ 0.1366 & -0.1441 & 1.4435 \end{bmatrix} \quad cov(:,:,3) = \begin{bmatrix} 370.9433 & 33.0306 & 8.5494 \\ 33.0306 & 13.3504 & -9.4842 \\ 8.5494 & -9.4842 & 26.6007 \end{bmatrix}$$

(a) K = 3 (b) K = 4

Figure 4: Segmentation results for the zebra image

K = 4

$$\mu = \begin{bmatrix} 36.3619 & -13.5228 & 20.3539 \\ 33.6922 & -13.6305 & 21.7129 \\ 50.8737 & -3.6655 & 13.5074 \\ 16.7898 & -5.2666 & 9.2837 \end{bmatrix} \quad \alpha = \begin{bmatrix} 0.3782 \\ 0.4541 \\ 0.0439 \\ 0.1238 \end{bmatrix}$$

$$cov(:,:,1) = \begin{bmatrix} 2.6739 & 0.0529 & 0.0066 \\ 0.0529 & 0.4771 & -0.0317 \\ 0.0066 & -0.0317 & 0.33722 \end{bmatrix} \quad cov(:,:,2) = \begin{bmatrix} 11.2947 & -0.1777 & 2.0833 \\ -0.1777 & 1.1421 & -0.1162 \\ 2.0833 & -0.1162 & 1.6248 \end{bmatrix}$$

$$cov(:,:,3) = \begin{bmatrix} 374.2589 & 39.5492 & -2.0951 \\ 39.5492 & 15.6164 & -12.0992 \\ -2.0951 & -12.0992 & 28.8502 \end{bmatrix} \quad cov(:,:,4) = \begin{bmatrix} 101.4013 & -48.5869 & 72.6403 \\ -48.5869 & 31.7449 & -41.4782 \\ 72.6403 & -41.4782 & 60.3826 \end{bmatrix}$$

K = 5

$$\mu = \begin{bmatrix} 36.1173 & -13.6955 & 22.9079 \\ 34.9034 & -13.5918 & 20.5618 \\ 5.9173 & 0.6163 & 0.5914 \\ 42.3423 & -2.0725 & 10.3208 \\ 27.9475 & -11.3565 & 18.4912 \end{bmatrix} \quad \alpha = \begin{bmatrix} 0.1674 \\ 0.6118 \\ 0.0432 \\ 0.0518 \\ 0.1258 \end{bmatrix} \quad cov(:,:,1) = \begin{bmatrix} 6.4789 & -0.4061 & -0.3089 \\ -0.4061 & 0.5579 & -0.0900 \\ -0.3089 & -0.0900 & 0.3612 \end{bmatrix}$$

$$cov(:,:,2) = \begin{bmatrix} 7.7215 & 0.2548 & -0.4223 \\ 0.2548 & 0.7031 & -0.0825 \\ -0.4223 & -0.0825 & 0.4525 \end{bmatrix} \quad cov(:,:,3) = \begin{bmatrix} 2.3392 & 1.1864 & -0.2680 \\ 1.1864 & 2.4123 & -0.8959 \\ -0.2680 & -0.8959 & 2.7564 \end{bmatrix}$$

$$cov(:,:,4) = \begin{bmatrix} 616.5155 & 5.0215 & 74.1338 \\ 5.0215 & 9.4630 & -7.0384 \\ 74.1338 & -7.0384 & 28.2103 \end{bmatrix} \quad cov(:,:,5) = \begin{bmatrix} 40.0390 & -8.3256 & 20.3584 \\ -8.3256 & 6.9366 & -7.8249 \\ 20.3584 & -7.8249 & 16.7240 \end{bmatrix}$$

As shown in Figure 2, 3 and 4, increasing the number of components gives different segmentation results. In our case, the cow and zebra image mainly contains 3 components: the grass field and the 2 colours on the animals. And thus using K = 3 creates less nosier results. However, in practice, the number of components in an image is unknown and might be difficult to determine. On the contrast, the mean-shift algorithm doesn't requires the knowledge of the number of components, but to determine a suitable radius is also tricky.