For Lab 1, my topic was image segmentation. I implemented three different clustering algorithms: FCM, Meanshift and Fuzzy Local Information C-Means. The following of this report will discuss the overview of these algorithms, how I implemented them and also the analysis of the results.

1. Overview of FCM, Meanshift and FLICM

- FCM

  FCM, developed by J.C.Dunn in 1973, and improved by J.C.Bezdek in 1981, is one of the most widely used fuzzy clustering algorithms. FCM algorithm attempts to partition a finite collection of n elements **X** = {X1, X2, X3·····Xn} into a collection of c fuzzy clusters with respect to some given criterion. Given a set of data, the algorithm returns a list of c cluster centres C = {C1,·····Cc} and a partition matrix W = wij∈[0,1], i = 1,·····,n; j = 1,·····,c, where each element wij is the degree to which element Xi belongs to clusrer Cj. Below is the pseudo-code of FCM in textbook:

  Let $X = \{x_1, x_2, \cdots\cdots, x_n\}$, where $x_k \in R^d$ is the set of vectors to be clustered.

  Initialization: Set
  C, the number of clusters desired
  m, the fuzzifier
  ε, the convergence threshold
  $V^{(0)} = \{v_1^{(0)}, \cdots\cdots, v_c^{(0)}\}$ an initial set of cluster centers
  Set t = 0
  PEPEAT
  DO FOR each k = 1,·····,n
  IF d(xk,vi) = 0 for some subset of clusters
  THEN
  Set $u_{jk}^{(t)} = 0$ for $j \notin I_k$ and $u_{jk}^{(t)} > 0$ for $j \in I_K$
  ELSE
  Compute $u_{ik}^{(t)}$
  ENDIF
  Set $t \leftarrow t + 1$
  Using $U^{(t-1)}$, estimate $V^{(t)}$
  UNTIL $\sum \| V_i^{(t)} - v_i^{(t-1)} \| < \varepsilon$

- MeanShift

  Mean shift algorithm, which presented in 1975 by Fukunaga and Hostetler, is a non-parametric feature-space analysis technique for locating the maxima of a density function. It is an iterative method and starting with an initial estimate x. Let a kernel function K(xi-x) be given. This function determines the weight of nearby points for re-estimation of mean. Typically a Gaussian kernel on the distance to the current estimate is used. $K(x_i - x) = e^{-c\|x_i - x\|^2}$. The weighted mean of the density in the window determined by K is $m(x) = \frac{\sum_{x_i \in N(x)} K(x_i - x) x_i}{\sum_{x_i \in N(x)} K(x_i - x)}$, where N(x) is the neighborhood of x, a set of points for which $K(x_i) \neq 0$. The difference between m(x) and x is called mean shift. Meanshift algorithm then sets $x \leftarrow m(x)$ and repeats until m(x) converges. Below is the procedure of Meanshift:

1.Randomly choose a point as center from points that have not been clustered.

2.Find the set M which consist of all points whose distance from center is within the bandwidth and assign these points to cluster C(each point can be assigned to many clusters)

3. Calculate the distance from center to every point in M and get the meanshift

4. move the center according to meanshift

5.repeat 2 3 4, until the meanshift is less than or equal to a threshold, and the center at the last iteration is the final center.

6. repeat 1 2 3 4 5 until all points are clustered (assigned to cluster)

7.For each point, set it to the cluster that it is assigned to most often.

1.3 Fuzzy Local Information C-Means

FLICM incorporates the local spatial information and gray level information in a novel fuzzy way. It can overcome the disadvantages of FCM and at the same time enhances the clustering performance. Compared to FCM, FLICM introduced a fuzzy factor $G_{ki} = \sum_{j \in N_i, i \neq j} \frac{1}{d_{ij}+1} (1-u_{kj})^m \|x_j - v_k\|^2$ and the objective function is :

$$J_m = \sum_{i=1}^{N} \sum_{k=1}^{c} \left[u_{ki}^m \|x_i - v_k\|^2 + G_{ki} \right].$$

The formulas to calculate U and V are:

$$u_{ki} = \frac{1}{\sum_{j=1}^{c} \left(\frac{\|x_i - v_k\|^2 + G_{ki}}{\|x_i - v_j\|^2 + G_{ji}} \right)^{1/m-1}} \qquad v_k = \frac{\sum_{i=1}^{N} u_{ki}^m x_i}{\sum_{i=1}^{N} u_{ki}^m}.$$

The procedure of FLICM is following:

1.Set the number C of the cluster, fuzzification parameter m and the stopping condition

2. Initialize randomly the fuzzy partition matrix

3. Set the loop counter b = 0

4.Calculate the V

5.Calculate U

6. repeat 4 5 until the stop condition.

2. Implementation

2.1 FCM

Firstly, I convert the RGB image to gray image and reshape it to a m * n * 1 array, where m is the first dimension and n is the second dimension of the gray iamge. Then I use this reshaped array as the input of my function called "FCM_Helper", which assign each pixel of the image to a cluster. This function returns a matrix called C and another matrix called dist. C contains all centers and dist contains the distance from each point to each center. I assign each point to the closest center. Then replace each point with the center For more detail about my implementation of FCM, please refer to my code.

2.2 MeanShift

The preprocess procedure is the same as FCM: convert RGB image to gray image and reshape it. Then I pass this reshaped array to the function MeanShift_Helper function. It returns a matrix called clustCent which contains all the cluster center, an array called point2cluster that contain the information which point belong to which cluster, and also a cell which contains the information about each cluster containing which cluster. In the MeanShift_Helper function, I randomly choose a point as center, and calculate its mean shift and move it. I repeat this step until convergence and record all the points that within the bandwidth in each iteration. Then I randomly choose another point as center and repeat the operations I descripted above until all the points have been assigned to at least one cluster. For more detail about my implementation of

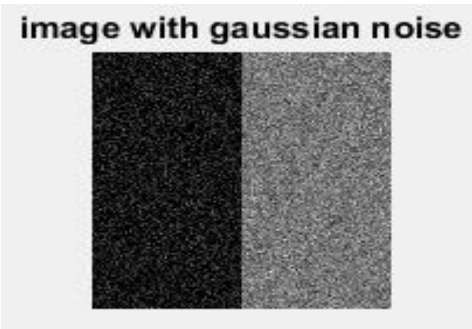
MeanShift, please refer to my code.

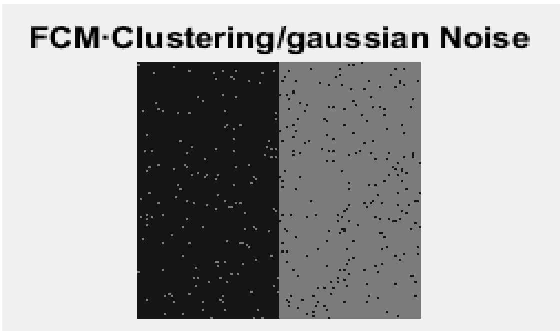
2.3 FLICM

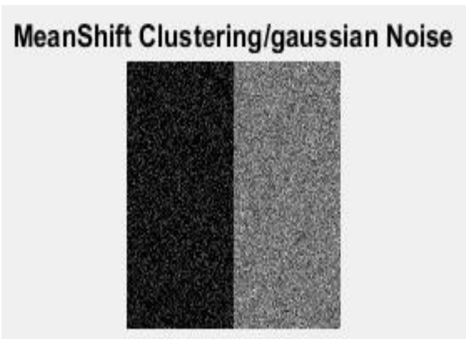
I preprocess the RGB image like above and then create windows for each point. For each point, we need a window to gather its' local information. The function createWindows returns a matrix called windows. It has width * width rows and m * n column, where width * width is the size of the window and m * n is the number of pixels. The function CreateWindows also return a one-dimension array called Distances, which contain the information that the distance between each point in the window to its' center. Then I call function FLICM_Helper, it returns the U matrix, V matrix, and iteration times. In the function FLICM_Helper, I initialize U matrix and then calculate the fuzzy factor G and then update V matrix. I repeat these operation until stop condition occurs. For more detail about my implementation of FLICM, please refer to my code.

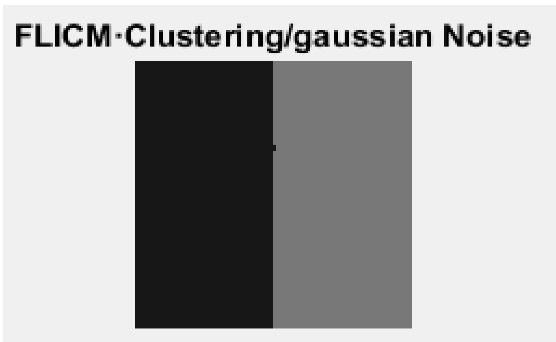
3. Result

In this experiment, I apply these algorithms to a synthetic test image corrupted by Gaussian, Uniform and Salt & Pepper noise, respectively. For FCM and FLICM, I set the cluster number to 2. The result shows that FLICM has the ability to overcome noise than FCM and MeanShift. Below is the cluster result:









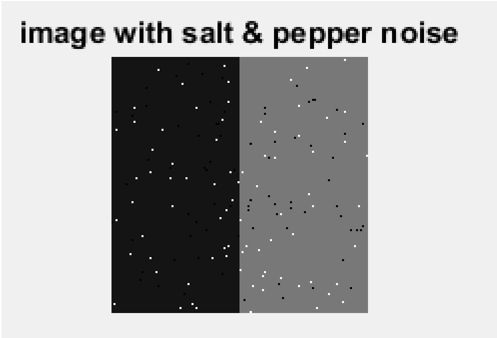
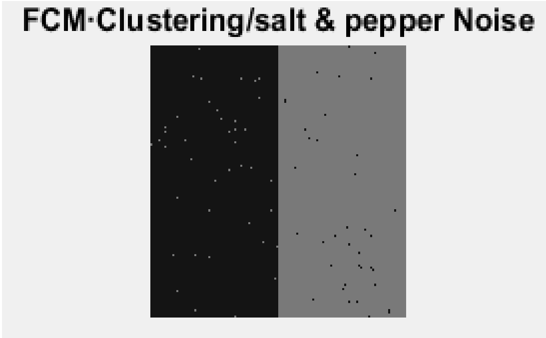
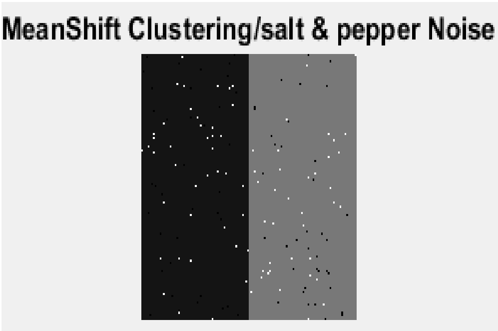
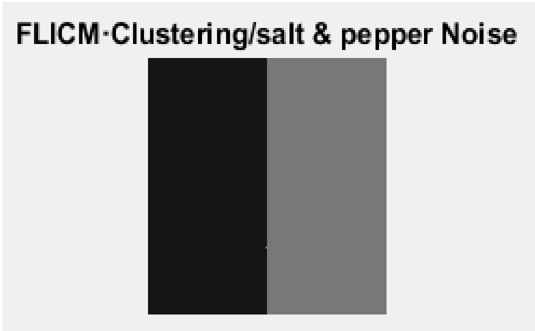


image with salt & pepper noise


FCM·Clustering/salt & pepper Noise


MeanShift Clustering/salt & pepper Noise


FLICM·Clustering/salt & pepper Noise