Introduction to Visual Information Processing CS455

Assignment 5

1) Author's Information

Ali Arda Eker

aeker1@binghamton.edu

2) Purpose

Performing Nearest Neighbor Classification and K-Means Clustering algorithms onto the given image with different parameters. Error rate is calculated. For the second part motion vectors are detected using given image frames.

3) Methods & Results

A) Pattern Recognition

First of all, I manually group the training vectors which are the upper half of the image into 3 clusters and got image M1. For each 4x4 block I calculated the average gray scale value and see if it is under 125, between 125 and 175 or above 175. If the block is under 125, it is in cluster 1 and labeled with class label 0. If a block is between 125 and 175, it is in cluster 2 and labeled with class label 128. If it is above 175, it is in cluster 3 and labeled with 255.

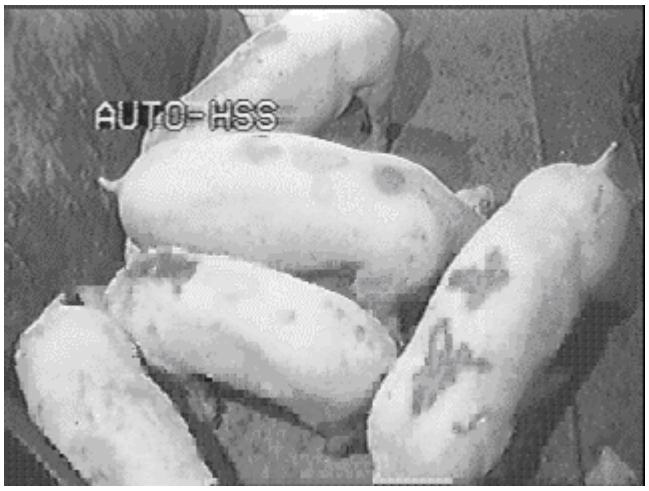
(M1)

After this I implemented Nearest Neighbor classification with different configurations. First I replaced each testing vector which is in the lower half of the image with the corresponding class label to produce N1. For doing this I calculated the distance between a training vector and each of the testing vectors and choose the smallest one. I used mean square error for calculating distance which is subtracting each pixel from its corresponding pixel, taking power of 2, adding them together and taking square root. So I did this operation for each test block and assigned the class label of the closest training block to the test block and got image N1.

(N1)

Notice: In the program to get N1, you first need to perform manually grouping because in that stage labels are set to a global vector to be retrieved later.

After this I replaced each testing vector with the corresponding training vector and got N2. To this I again used mean square error to find the closest match but I didn't assign its label this time instead I replace each pixel of the testing block with each pixel of the training block found.

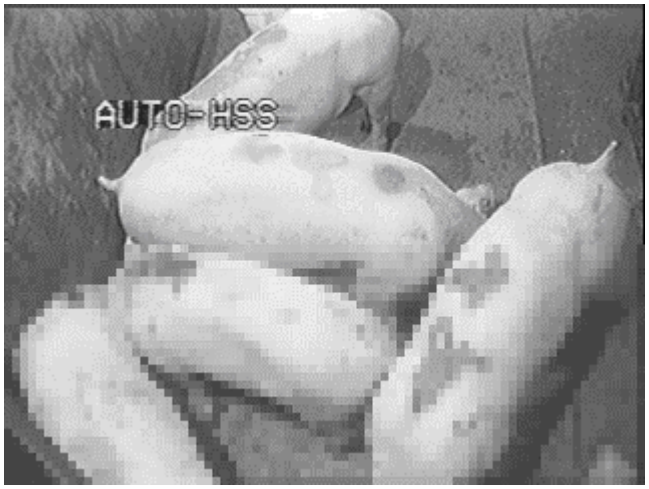


(N2)

You can see the change in test data. Block effect is not as much as the N1 because I replaced test blocks with train blocks thus all of their pixels are replaced but for N1, I replaced just class label.

Notice: In the program to get N2, you should not perform manually grouping before because program needs original training data.

After this I used average value of the corresponding training vector to replace with the test blocks and got N3. Corresponding training vector again means the closest match in terms of the mean square error



(N3)

Since I used the average value of closest training blocks and assign it to the targeted test block the block effect is more than N2.

Notice: In the program to get N3, you should not perform manually grouping before because program needs original training data.

After this, I used the average value of the corresponding training vector to replace with the testing blocks to get N4. This means when I found the closest block in terms of mean square error, I looked which class it belongs and took the average of the whole class. This is done by calculating each pixel of each block of the associated class and diving the total number of pixel the class has.

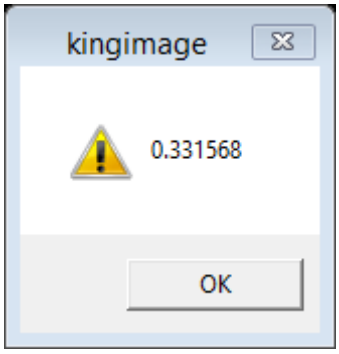
(N4)

Notice: In the program to get N4, you should perform manually grouping first because the classes are constructed in that stage.

Next step is performing manual grouping to the second half of the image to compare it with the N1. This means how well our classification algorithm is trained and how accurate it is. T1 is created using same procedure with the M1 but it is applied to the test blocks.

(T1)

To compare T1 and N1, I used mean square error and measure error rate. This is done by subtracting each pixel in N1 from its corresponding pixel in T1, taking their power of 2, adding them together, taking square root and then dividing the number by the total number of pixels. So the error rate E is 0.33.



Notice: To calculate error rate, you first need to manually group the upper part, then perform N1, then close the current file and while the program is still running open the same image and manually group the lower part, and then you can measure error rate. This is so because N1 fills a global variable named valuesN1 which holds the N1 values of the testing blocks and the program needs original image to manually group the lower half.

Next step is performing K-Means Clustering. The algorithm starts by choosing 3 random initial cluster centers. It needs to have 3 centers because it is known that the image is divided into 3 clusters. After this I assigned each block to the associated cluster using the same distance concept again. After filling the clusters I updated the centers by calculating the average gray scale value of each cluster and choosing the block which has the closest average value to the class average. I repeated this until the image is clustered well enough to detect the pigs. 5 iteration was enough to get image K1.

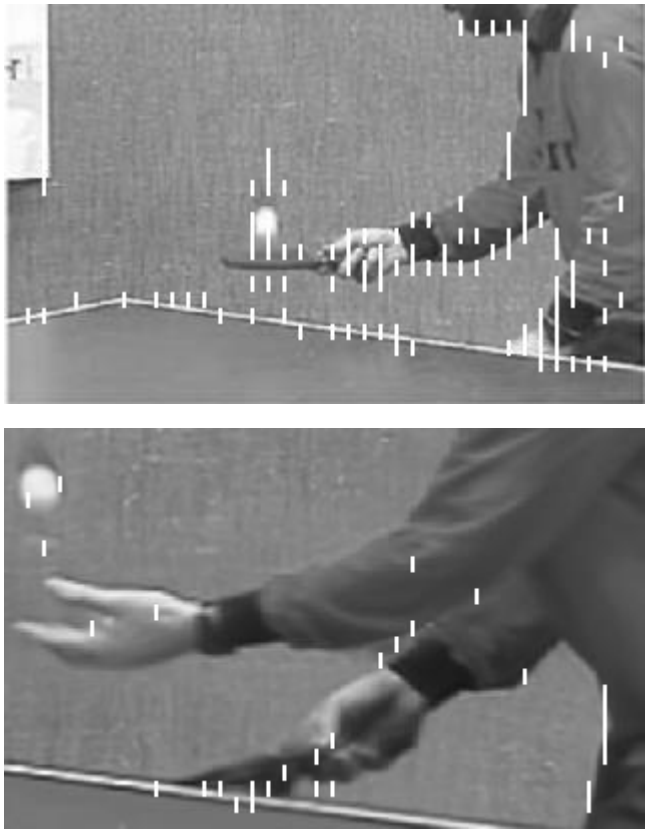
(K1)

To perform K-Means Clustering I implemented a structure to abstract the pattern vectors which made program much more efficient. The pattern objects have a 2 dimensional vector to represent the pixel values in a block, a double typed variable avg to hold the average value which is used to update cluster centers, a Boolean variable named center to indicate whether a block is chosen as center or not, and a integer cluster variable which indicates which cluster it belongs.

B) Motion Compensation and Tracking

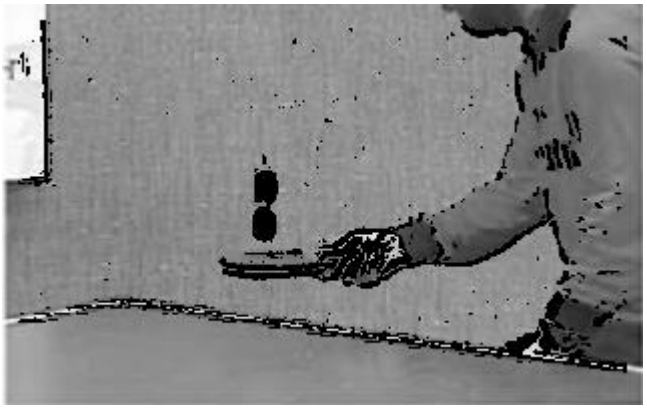
To create motion vectors and calculate difference between frames I implemented a structure named motion pattern which is composed of an 8 by 8 vector and average variable. I used this structure to retrieve pixel information faster.

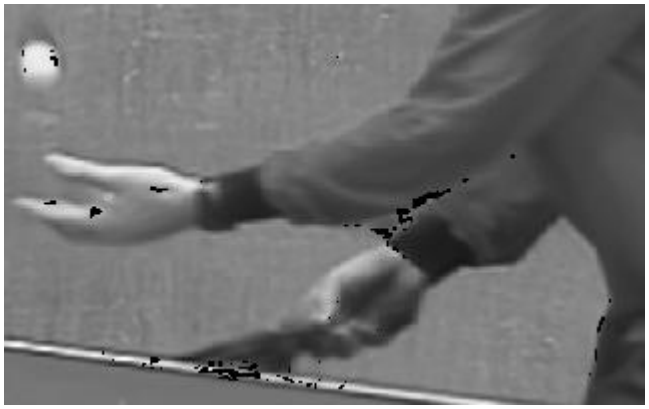
To detect and draw motion vectors I compared 2 frames and calculated change in average gray scale values for each block. If a change is found I put 255 to the corresponding pixels to indicate a motion vector as shown below.



I used 2 global variable named frame1 and frame2. Those variables are 2 dimensional vectors which hold motion pattern structures. In each index of those vectors I put a block which has an average value and corresponding pixel values. This made my program faster and more robust.

To calculate difference between frames I compared the image sequence and simply subtract each pixel one by one. I put a threshold value 20 to detect major differences and assigned value 0 to indicate change.





Notice: To get shown images using the program you first open first frame and click retrieve frame 1 then close the file and open second frame, click retrieve frame 2 then you can click motion detection or extract difference. But If you performed motion detection close the program and open it again to do extract difference.