

**Introduction to Computer Vision**

**Coursework**

**Submission 2**

**Your name \_\_\_\_\_\_\_\_ HAO BAI \_\_\_\_\_\_\_\_**

**Student number \_\_\_\_\_\_\_\_201218765\_\_\_\_\_\_\_\_\_\_**

**Question 4(a)**

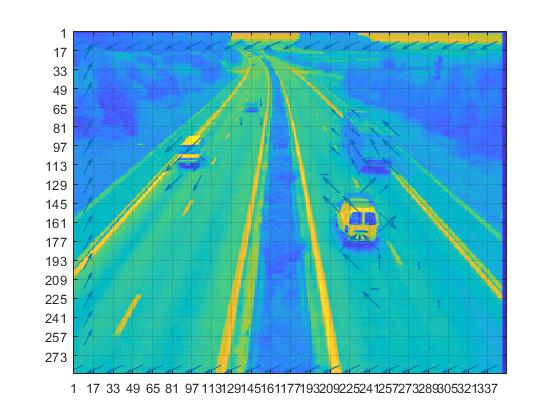
It+1



It



Motion field of It+1



**Question 4(b)**

Pt+1



It+1

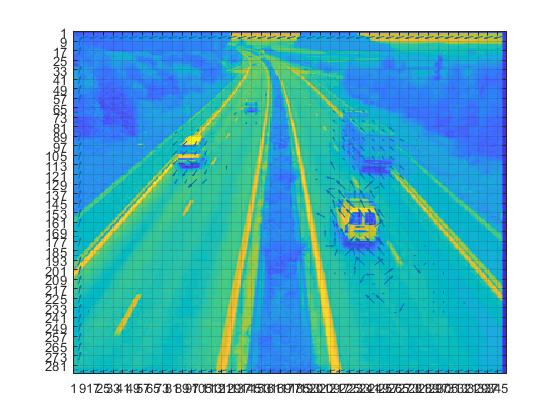


**Your comments**

**Question 4(c)**

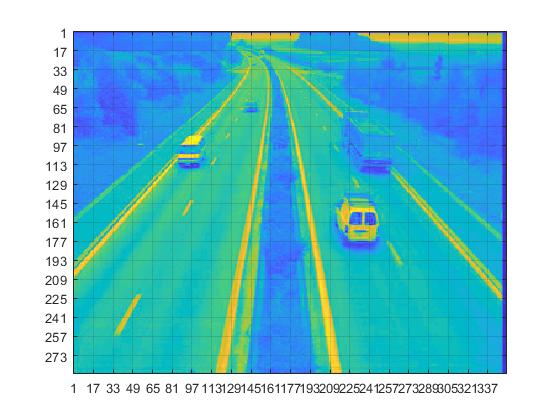
Pt+1

Block size = 8x8



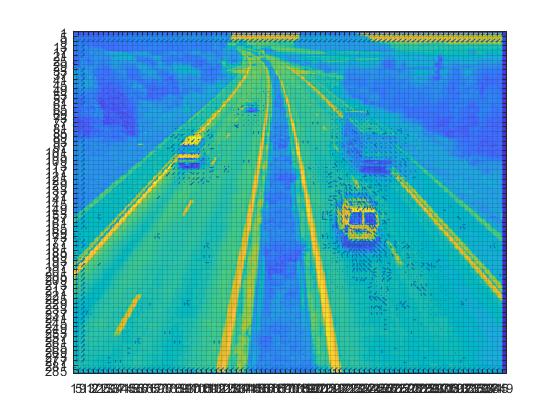
Pt+1

Block size = 16x16



Pt+1

Block size = 4x4

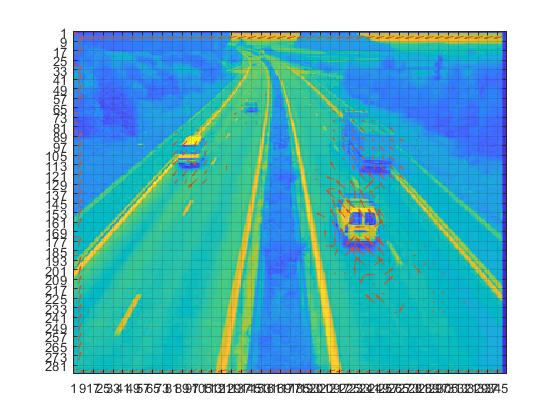


**Your comments:**

**Question 4(d)**

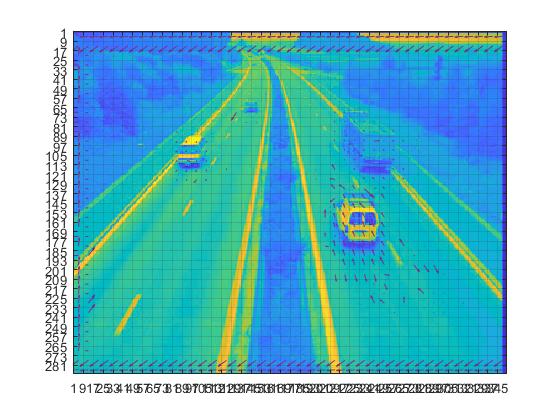
Pt+1

Window size = 16x16



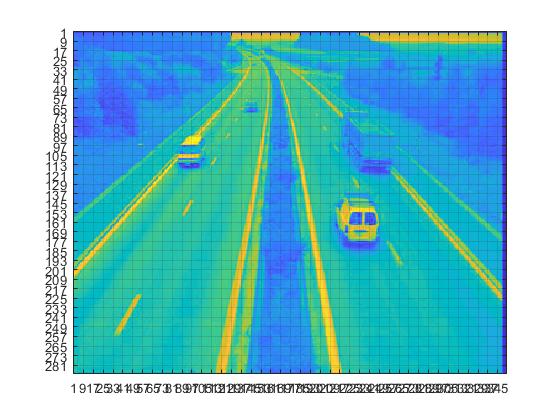
Pt+1

Window size = 32x32



Pt+1

Window size = 8x8



**Your comments:**

**Question 4(e)**

**Plot graphs:**

Time versus window size

Time versus block size

**Your comments:**

**Question 5(a)**

**Original frames:**

Selected frame 2



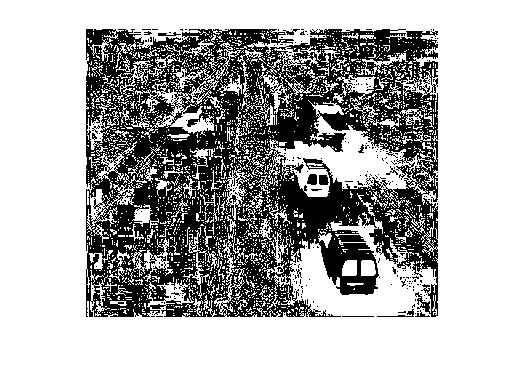
Selected frame 1

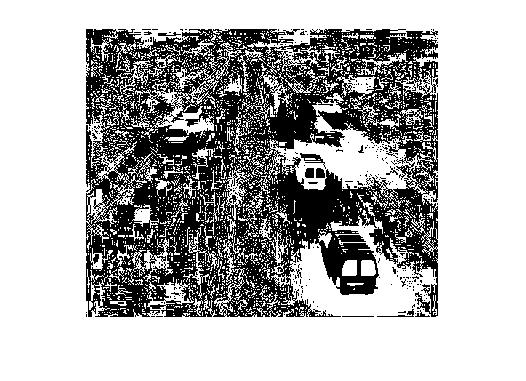


Reference frame

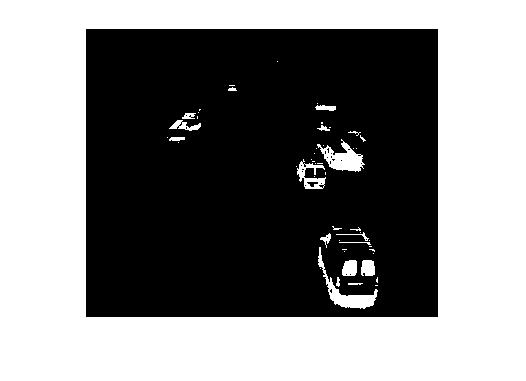


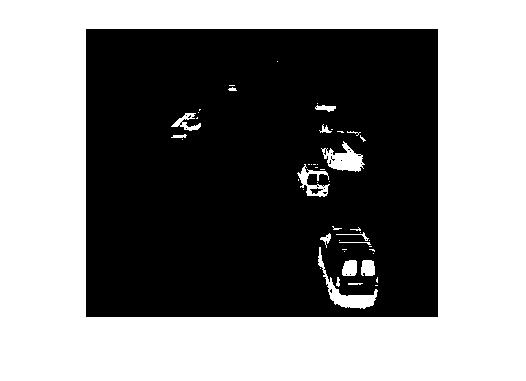
**Frame differencing:**





**Threshold results:**





**Your comments:**

**Threshold = 40**

**Question 5(b)**

**Original frame:**

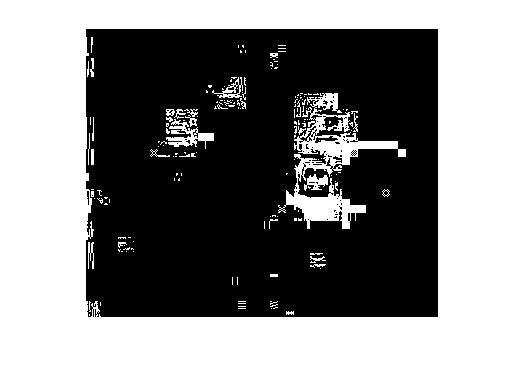
Selected frame 1

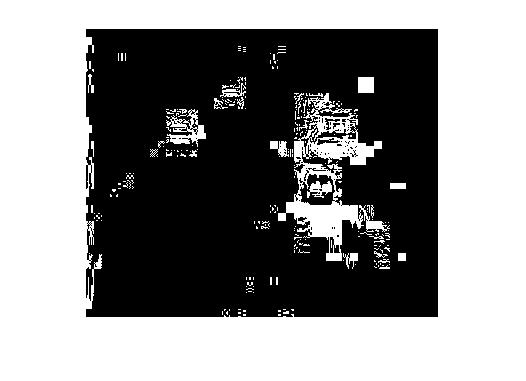


Selected frame 2

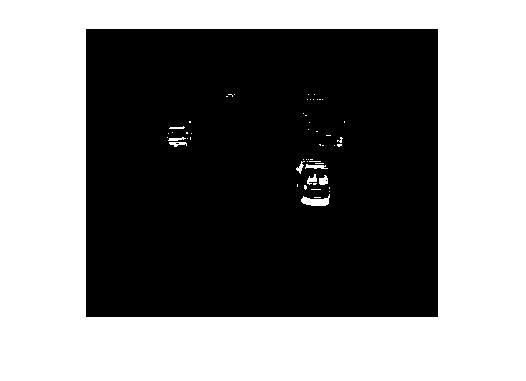


**Frame differencing:**





**Threshold results:**





**Your comments for 5a,5b:**

**Question 5(c)**

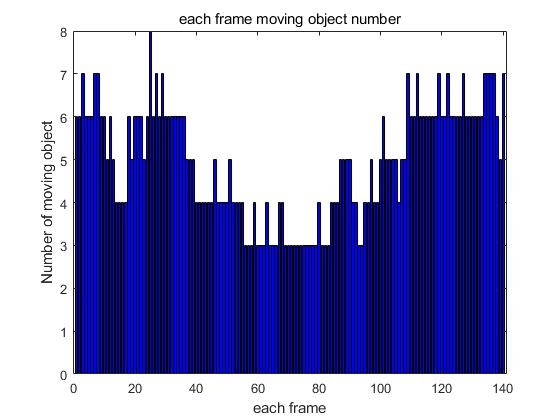
Generated background



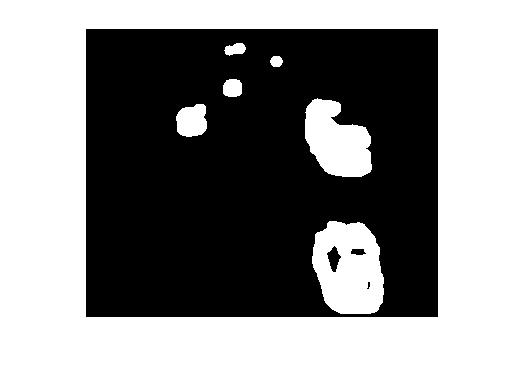
**Your comments:**

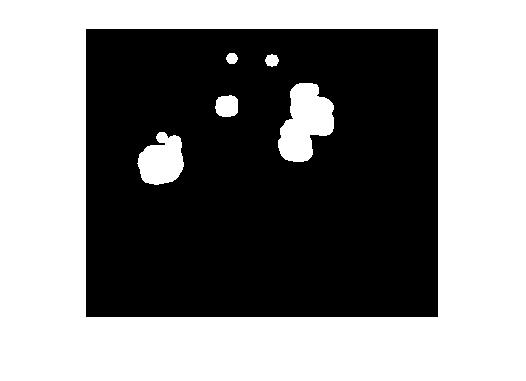
**Question 5(d)**

Bar plot



**Your comments:**

****

****

**Question 6(a)**

**Three non-consecutive windows**

W3



W2

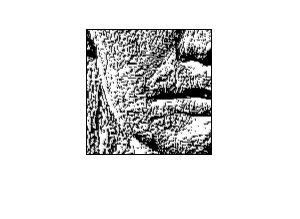


W1

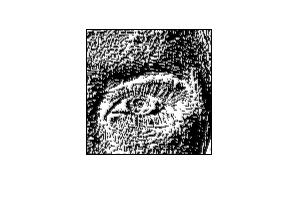


**LBP of windows**

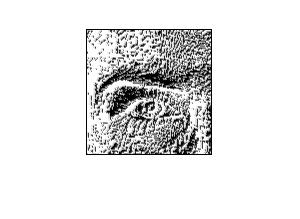
LBP3



LBP2

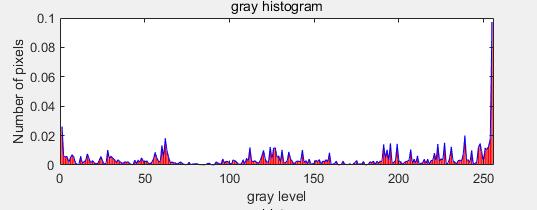


LBP1

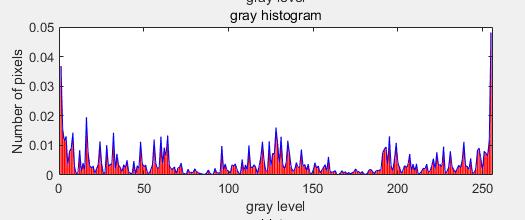


**Histograms of LBPs**

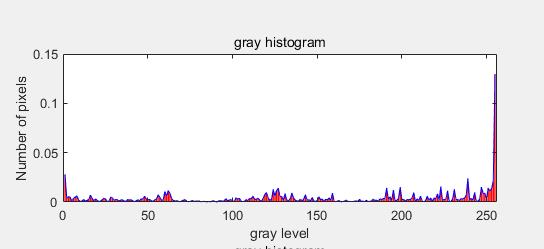
H3



H2



H1



How to show the result:

Run the ***cell 6- a*** in ***ICV\_Assignment2\_main\_file.m*** file.

The algorithm in this cell is that:

1. Load the face image(*face-2.jpg*) and convert to grayscale mode using *rgbgray* function
2. Set the *Block size* = 128, that means each block windows is 128 pixels \* 128 pixels. That divided the face image into 4 small windows.
3. Using *ICV\_divideIntoNonOverlapping* functions. That required input an *image* and the *block size.* And return a group of non-overlapping images that divided from the images.
4. Using *size ()* function to get how many blocks, the image has been divided.
5. The loop is to read each blocks and handle.
6. The *ICV\_LBPfunction* is a function that required input a image and return a LBP image.
7. Then function *ICV\_hisgram* will print the histogram of the frequency of the numbers, the required input is the block images and number of the all block images and the current count number. That will help the function draw the bar chart together.
8. Each of the histogram will as the feature descriptor of each block.

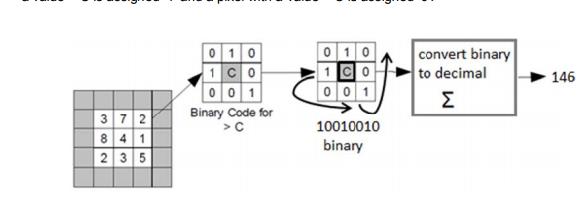
The algorithm of the using functions:

*ICV\_divideIntoNonOverlapping:*

1. This function will divide the big image into small piece of the non-overlapping images as the return value accordong to the block size needed.
2. The function will get to know the length and width of the image
3. sometimes the image cannot cut into equally sized non-overlapping block as the asked size, the function will cut extra length or width.
4. The function will divide the image into small block, the step size is the block size. Each block will have stored in the return value.
5. The block size is 128 pixels, so the return value will be a 128\*128\*4 matrix.

*ICV\_LBPfunction*

1. This function compares 8 of each pixel neighbors with the pixel self. If the number larger or equals to the pixel, then the neighbors number equals to 1. Else, the neighbors number will equals to 0.
2. And then each pixels equals to the numbers that converted counter-clockwise neighbors binary to decimal number.



*ICV\_LBPfunction*

1. The function will count the of the gray level and the total number of the pixels.
2. The function will calculate the frequency of each numbers and using bar chart to show.
3. Normalize the histogram and show in the same sub-graph.

**Question 6(b)**

**Two example images:**

Car image

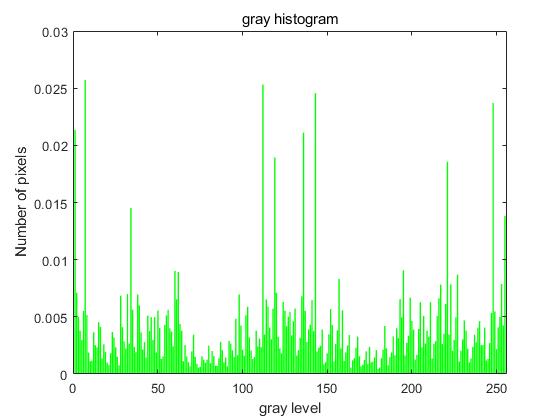


Face image

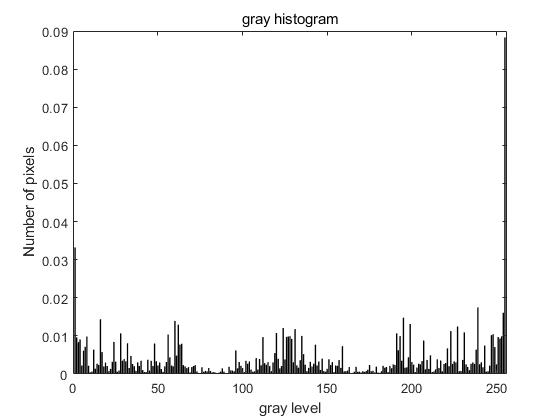


**Descriptors:**

Car descriptor



Face descriptor



**Your comments:**

How to show the result:

Run the ***cell 6- b*** in ***ICV\_Assignment2\_main\_file.m*** file.

The algorithm in this cell is that

1. Load the face and car image (*face-2.jpg and car-1.jpg*) and convert to grayscale mode using *rgbgray* function
2. Set the *Block size* = 128, that means each block windows is 128 pixels \* 128 pixels. That divided the face image into 4 small windows.
3. Using *ICV\_divideIntoNonOverlapping* functions. That required input an *image* and the *block size.* And return a group of non-overlapping images that divided from the images.
4. Using *size ()* function to get how many blocks, the image has been divided.
5. The loop is to read each blocks and handle.
6. The *ICV\_LBPfunction* is a function that required input a image and return a LBP image.
7. Then function *ICV\_hisgram* will print the histogram of the frequency of the numbers, the required input is the block images and number of the all block images and the current count number. That will help the function draw the bar chart together.
8. Each of the histogram will as the feature descriptor of each block.
9. Combine each of the sub-histogram in to one histogram that is the descriptor of the whole image.
10. To combine the frequency of each of the number occur, the function will add all of the frequency first and divide by the number of the blocks.

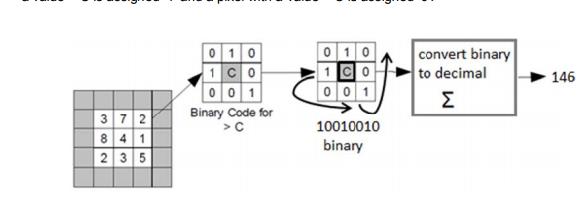
The algorithm of the using functions(same as the 6 -a):

*ICV\_divideIntoNonOverlapping:*

1. This function will divide the big image into small piece of the non-overlapping images as the return value accordong to the block size needed.
2. The function will get to know the length and width of the image
3. sometimes the image cannot cut into equally sized non-overlapping block as the asked size, the function will cut extra length or width.
4. The function will divide the image into small block, the step size is the block size. Each block will have stored in the return value.
5. The block size is 128 pixels, so the return value will be a 128\*128\*4 matrix.

*ICV\_LBPfunction*

1. This function compares 8 of each pixel neighbors with the pixel self. If the number larger or equals to the pixel, then the neighbors number equals to 1. Else, the neighbors number will equals to 0.
2. And then each pixel equals to the numbers that converted counter-clockwise neighbors binary to decimal number.



*ICV\_LBPfunction*

1. The function will count the of the gray level and the total number of the pixels.
2. The function will calculate the frequency of each numbers and using bar chart to show.
3. Normalize the histogram and show in the same sub-graph.

**Question 6(c)**

**Block diagram of classification process**

**Your comments:**

**Question 6(d)**

**Your comments:**

****

**Question 6(e)**

**Your comments:**

**Question 6(f)**

**Your comments**