Queen Mary
University of London

**Introduction to Computer Vision**

**Coursework**

**Submission 2**

**Your name** Yinghao Qin

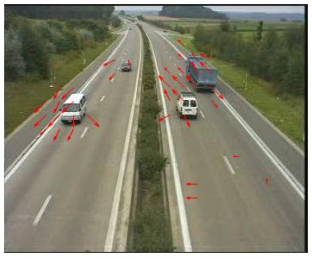
**Student number** 190189237

**Question 4(a)**

$I_t$

$I_{t+1}$

Motion field of $I_{t+1}$



**Question 4(b)**

$I_{t+1}$

$P_{t+1}$

**The predicited frame**

**Your comments**

**For the motion field:**

**1. The steps to get the motion field. Firstly, I used the Gaussian filter to filter the images. The next step is that the frames are divided into equally sized non-overlapping blocks and the position information of each block is returned. After that, dealing with the next frame, the corresponding block is found on the search window which makes the error measure the least. Finally, pick the point in the top left corner as the representative of the block, calculate the velocity of each block and draw the motion vector on the next frame.**

**2. For the boarder problem, padding is not a good method for this task, for example mirror and replicate will change the real motion in the image, and zeros method does not contribute to the block matching task, therefore I do not process the outermost block in the image.**

**3. In the motion filed image, we can find that there are some arrows on the road (background). This may be because there are noises on the area.**

**For the predicted frame:**

**1. Comparing the two images, in general the effect of the prediction is good, but we can find that there are some missing details or incorrect prediction in the predicted image, for instance the black line of the leftmost vehicle.**

**Question 4(c)**

$P_{t+1}$

Block size = 4x4

$P_{t+1}$

Block size = 8x8

$P_{t+1}$

Block size = 16x16

**Your comments:**

**When using a $16 \times 16$ search window, with the increasing of block size, a larger block is moved.**

**1. When block size equals 4 by 4, there are too many small blocks, and more block will be moved just like noise.**

**2. When block size equals 8 by 8, the image look gets neater, but still many moved blocks.**

**3. When block size equals 16 by 16, the predicted image is the neatest.**

**Question 4(d)**

$P_{t+1}$

Window size = 8x8

$P_{t+1}$

Window size = 16x16

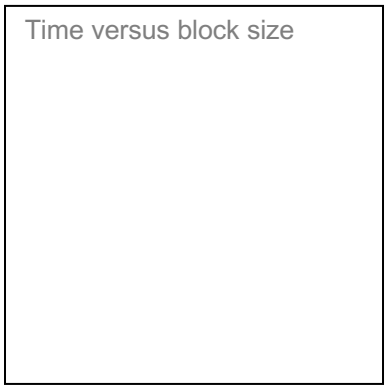
$P_{t+1}$

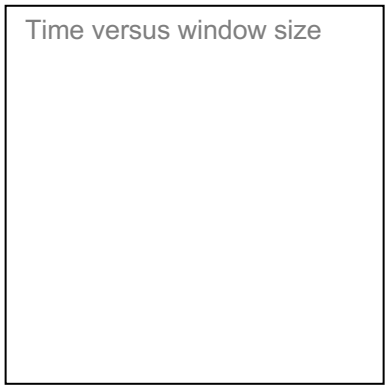
Window size = 32x32

**Your comments:**
**When using a $8 \times 8$ block, with the increasing of window size, a larger block is moved.**

**Question 4(e)**

**Plot graphs:**




**Your comments:**

**Question 5(a)**

**Original frames:**

Reference frame

Selected frame 1

Selected frame 2

**Frame differencing:**

**Threshold results:**

**Your comments:**

**1. In order to denoise, I used the Gaussian kernel of 3 by 3 to filter these frames. After that, I selected 2nd and 100th frame as the sample. Besides, I set the threshold as 20.**

**2. From frame differencing, the two images are dark, they are difficult to be recognized but we can still recognize the outline of vehicle. After the subtraction of the reference frame and selected frame, the pixel value on the unchanged area (background) is close to 0, and the pixel value on the changed area (such as moving car) is close to a bigger number.**

**3. Comparing the differencing frames and threshold results, the threshold results are lighter. Because the threshold processing enhances the contrast.**

**4. From the threshold results, it is easy to recognize the vehicle (moving objects). In addition, the right image has two cars on the bottom right-hand corner, but the corresponding original frame only has one car on the same area. This is because the absolute value is calculated between the reference frame and selected frame, so the cars on the refence and the cars on the selected frame will all show up.**

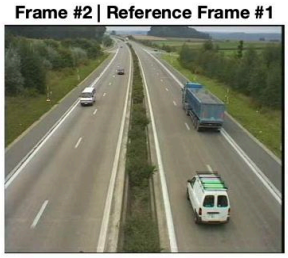
**5. For pick of the threshold value, when the threshold is too small to denoise, the image will look messy, and when the threshold is set big the image will lose many details.**

**Question 5(b)**

**Original frame:**

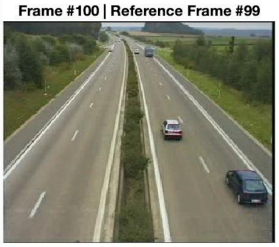
Selected frame 1

Frame #2 | Reference Frame #1



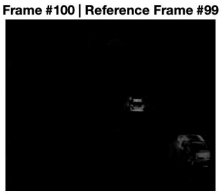
Selected frame 2

Frame #100 | Reference Frame #99



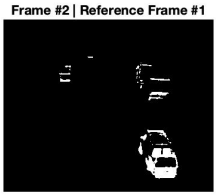
**Frame differencing:**

Frame #100 | Reference Frame #99

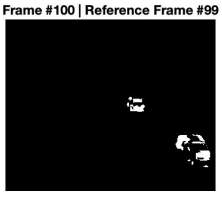


**Threshold results:**

Frame #2 | Reference Frame #1



Frame #100 | Reference Frame #99



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

**1. I selected 2nd and 100th frame as the test samples, and their reference frames are 1st frame and 99th frame respectively. Besides, the threshold is still 20.**
**2. The left images are totally the same as those on 5a, because they select 2nd frame and the 1st frame is their reference frame.**
**3. For the right differencing frame, it changes a lot comparing with that in 5a. First, the number of recognized cars becomes smaller (the black area becomes larger). This is because, for the two consecutive frames, their backgrounds are more stable i.e. the change in the image is smaller. Whereas for the differencing frame in 5a, it is messy and has larger white area on account of the bigger change between 100th frame and 1st reference frame.**
**4. For the right threshold result, it is obvious that there are less vehicles comparing with that in 5a. Because the absolute result will be calculated between the two frames, so the result will reflect the difference between the two images. For example, the car in threshold results in 5b may be stretched and there are more cars in 5a than the original image.**

**Question 5(c)**

Generated background

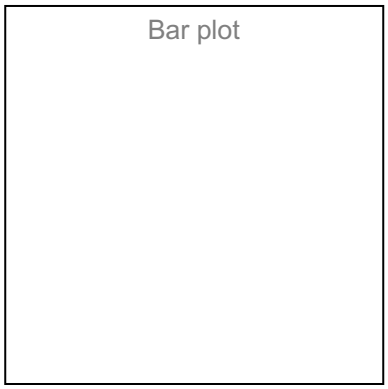
**Your comments:**

**1. The steps to get the reference frame (background). Firstly, repeat the method in 5b to get the threshold results of all the consecutive frames. Then, judge whether each pixel value of these threshold results is 0, if so count and accumulate each pixel for all the frames; do nothing otherwise (0 implies the area is unchanged i.e. background, otherwise the area is changed i.e. moving objects). Finally, for each pixel, we divide the value of the accumulation by the corresponding number, and then we get the reference frame.**

**2. The effect of the result is pretty good - there is no vehicle (moving objects) on the roads.**

**3. The generated background can be used further for example extracting the objects. And the method implementation is easy.**

**Question 5(d)**




**Your comments:**

**Question 6(a)**

**Three non-consecutive windows**

W1

W2

W3

**LBP of windows**

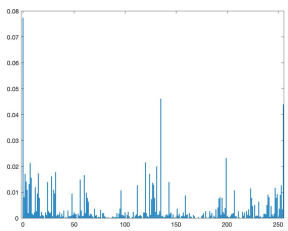
LBP1

LBP2

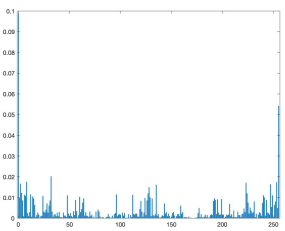
LBP3

**Histograms of LBPs**

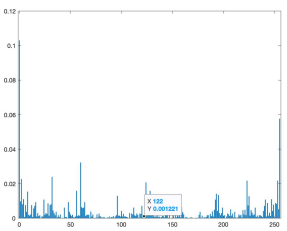
H1



H2



H3



**Question 6(b)**

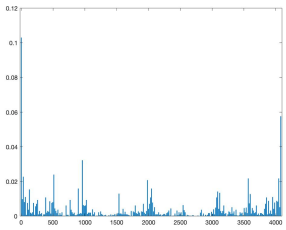
**Two example images:**

Face image

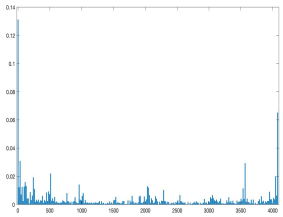
Car image

**Descriptors:**

Face descriptor



Car descriptor



**Your comments:**

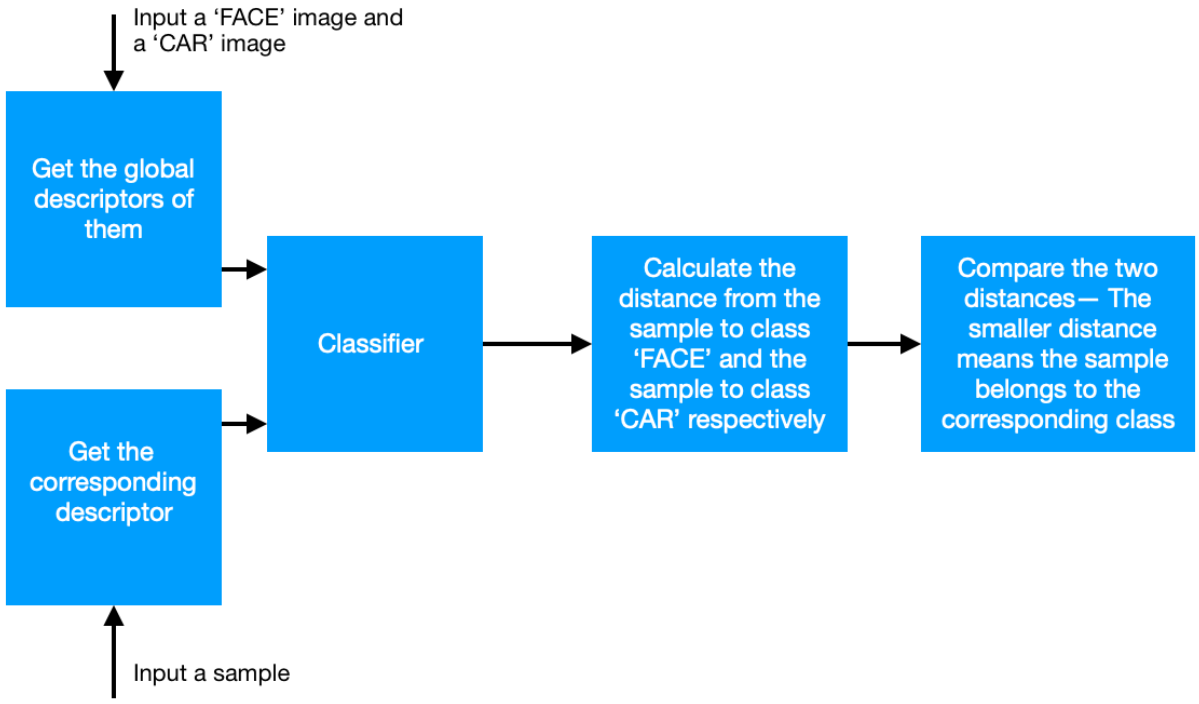
**1. The steps below about how I got the global descriptor for the image. Firstly, I divided the input image into 16 equally sized windows. On the next step, the point is to get the LBP image corresponding to each window, so I handled the boarder problem for each of these windows with mirror padding technique and then used LBP operating on these windows. After that, with processing these LBP images I got these corresponding feature descriptors (histograms). Finally, all these local descriptors were combined into a global descriptor.**

**2. Comparing the face descriptor with the car descriptor, it is clear that there are a lot of differences in the two descriptors such as the summits and intensity. It is obvious that the different kinds of image will have the mostly different descriptors.**

**3. We can still recognize some shapes from the LBP images.**

**Question 6(c)**

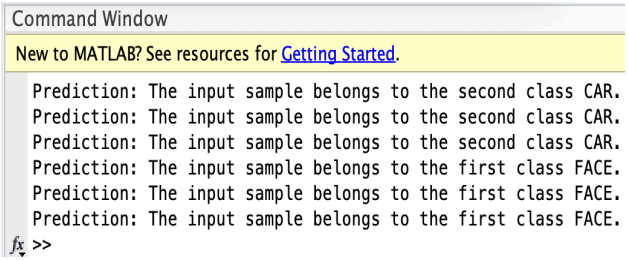
**Block diagram of classification process**

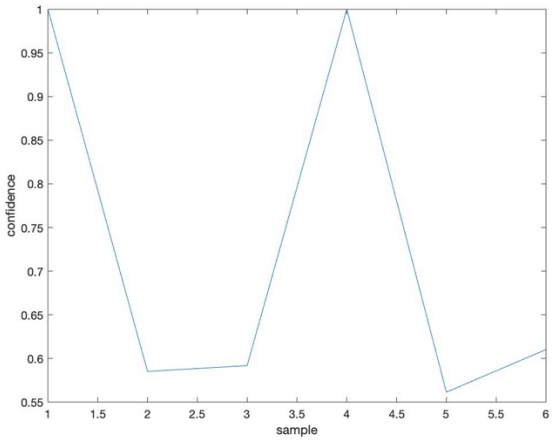



$$Distance = \sum \sqrt{(p-q)^2}$$

**$p$ denotes each value of the class image descriptor, and $q$ denotes each value of the sample image descriptor.**

**Your comments:**



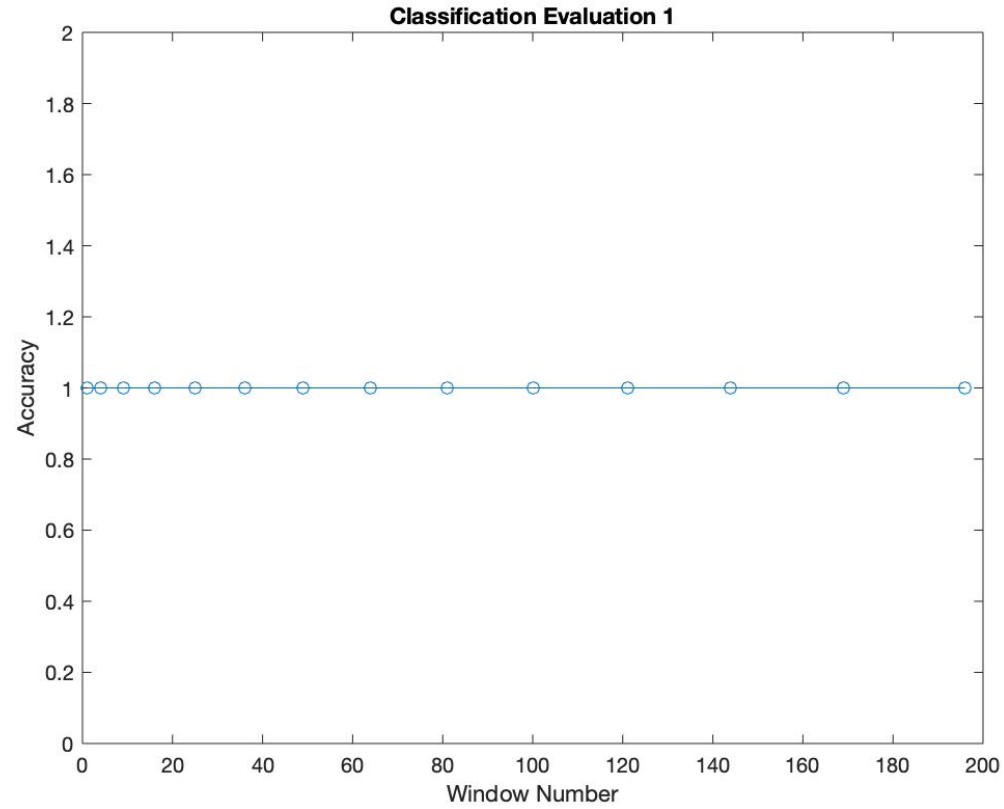



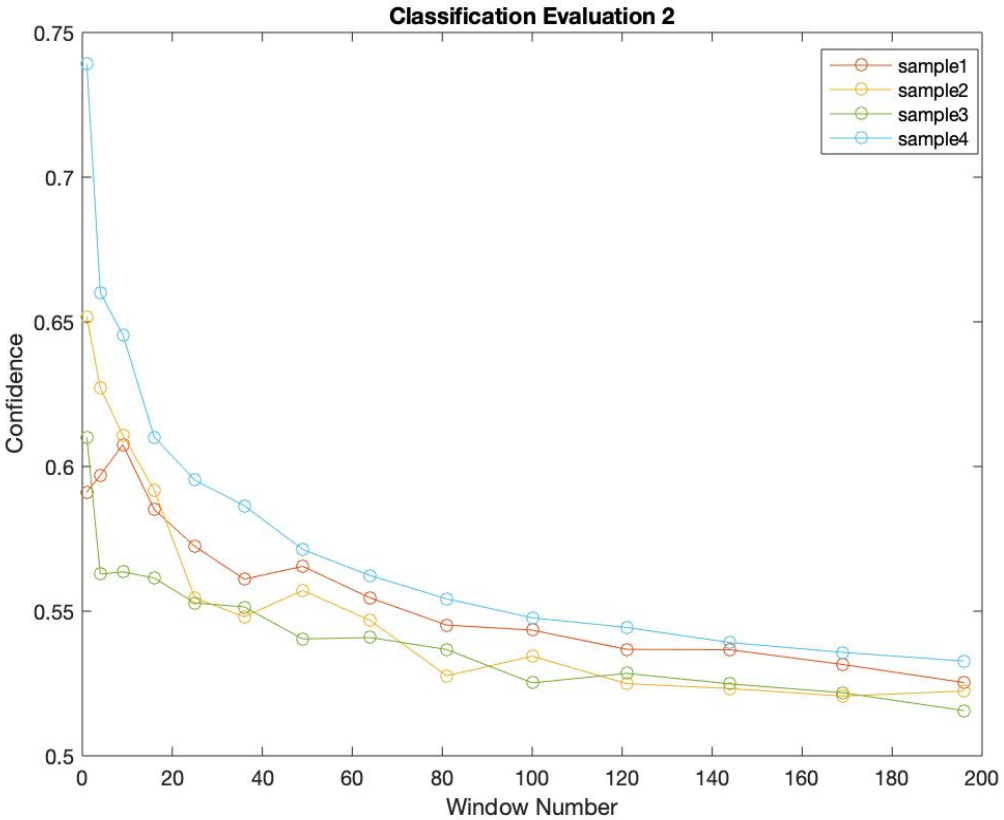

**1. From the left image, the prediction result is surprisingly accurate. In this case, we have known that the previous three samples are Car and the latter three samples are Face. And the prediction results are totally right. all the samples are inputted into the classifier,**

**2. From the right image, the confidence for the 1st and 4th sample are 1, because the 1st sample is selected as the class 'CAR' and 4th sample as the class 'FACE' of the classifier. Besides, the confidences are above 0.5, which means our prediction result is accurate.**

**Question 6(d) (e)**

**Your comments:**



**Question 6(d)(e)**

**Your comments:**

**For testing, I set 'car-1.jpg' and 'face-1.jpg' as the classes for creating the classifier and the rest images of the provided dataset were for testing.**

**For the first image:**

**1. it is clear that the prediction accuracy is always 1 no matter how many the window number changes. In other words, the classifier can correctly classify the test samples in the range from 1 to 196.**

**For the second image:**

**1. With the window size decreasing (the window number increases), the confidence decreases as well for all test samples. This is probably because say the window size is too small, the local descriptor cannot represent the feature of this window. For example, for the eyes of the 'Face' image, when the window size is too small, these windows cannot represent the intact feature of the eyes.**

**2. With the window size increasing (the window number decreases), the confidence generally increases for all test samples, which means the classification quality is great.**

**3. In general, the larger the window, the more representative the image features are. However, we should not select a too large window as well, because for a too large window a lot of feature details will be ignored, and it is also difficult to avoid the situation which different images have similar LBP descriptor. To summarize, a suitable window size should be selected to classify images.**

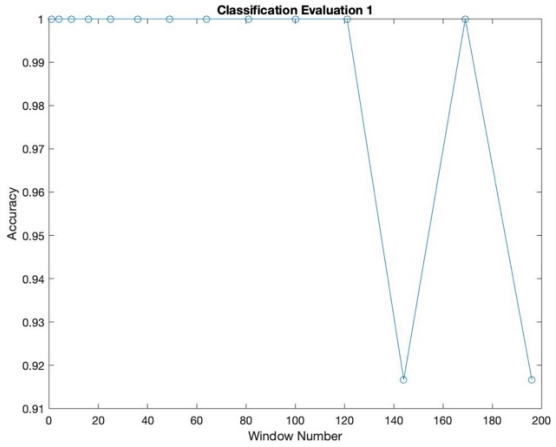
**In addition, I also used an extended dataset ('TestDataset') to test the prediction result, but we need to modify some parameters.**

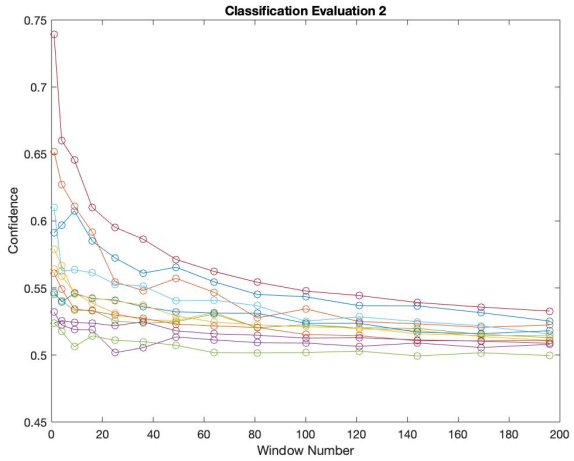
**In the function – ICV_Main**

**Line 137: for j <= 5**

**Line 146: for j <= 5**

**The prediction result is shown as below:**




**We can find that when the window number is 149, there is a mistake and the corresponding sample confidence is below 0.5.**

**Question 6(f)**

**Your comments**

**In order to analyse dynamic texture, we could use block matching technology and LBP to extract the motion and appearance features of dynamic texture. However, there will be limitations, for example firstly, the loss of texture intensity change information leads to incomplete texture feature extraction since only a single LBP Operator is used for calculation; secondly, LBP is used as a matching criterion in motion feature extraction, which leads to multiple matching points, thus affecting the matching accuracy.**