Introduction to Computer Vision

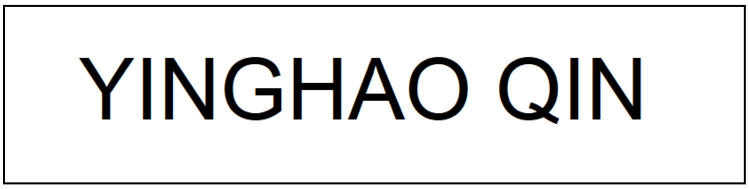
Coursework

Submission 1

Your name Yinghao Qin

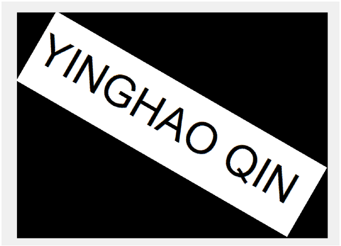
Student number 190189237

Question 1(a):

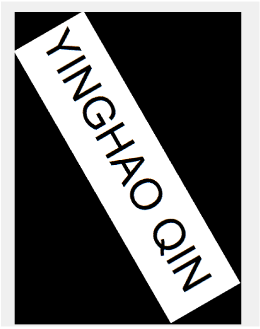


Rotated images:

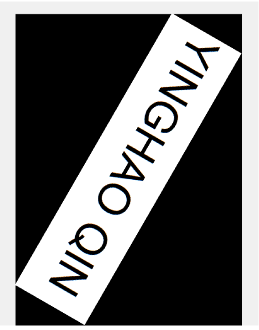
θ = 30 deg



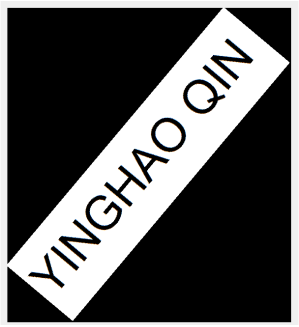
θ = 60 deg



θ = 120 deg

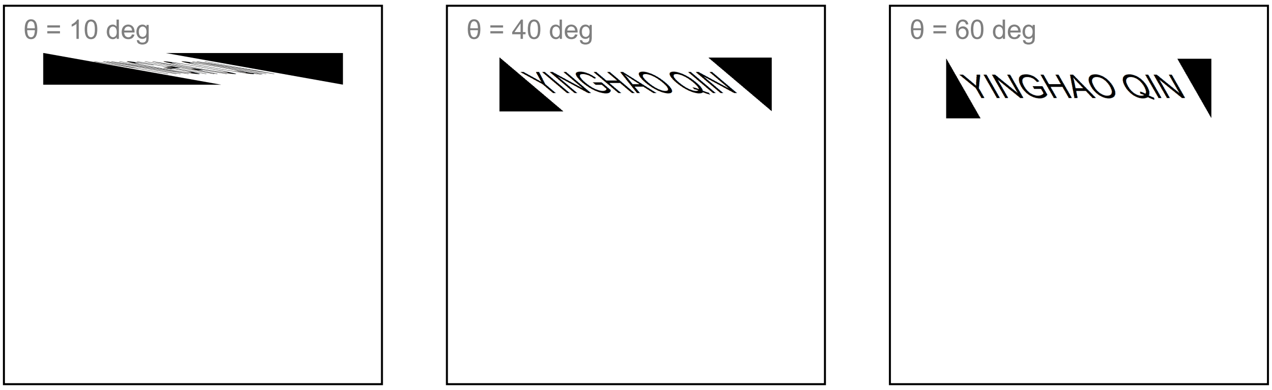


θ = -50 deg



Fill the available spaces for your answers. Do not extend the spaces and do not change the formatting of the pages. Use the same font size and the same number of lines as in the original file

Skewed images:



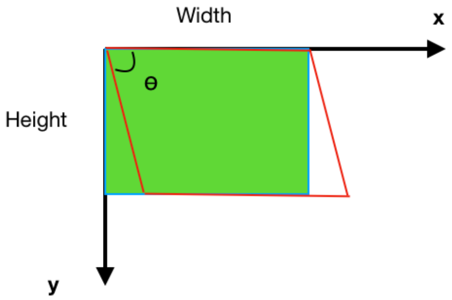
Your comments:

In this case, I used forward mapping and bilinear interpolation. And the skewed images are the results of skewing the original images along the y-axis.

It is clear that the rotated images still have 'black points', but the there are no 'black points' in skewed images. This is because rotating the images leaves some pixels undefined, but skewing the images means skewing the coordinate axis. In addition, bilinear interpolation cannot be used to eliminate noise.

Rotated images:
1. The steps I implemented the operations: First of all, I calculated the size of new image by the geometric knowledge and set up a new image. Then, compound transformations were needed to multiply each pixel of the original image, and the compound transformations contain three steps - translate the image to the center of the coordinate, rotate it, and finally translate it back to the original area. After that, I used the inverse mapping for the implementation of transformation. Finally, I used the Nearest Neighbour Interpolation to solve the possible noises.
2. In addition, I also listed the alternative parts such as the forward mapping of transformation and Bilinear Interpolation.
3. When computing the size of new image, we need to pay attention to the trigonometric functions of degrees in case the results are negative.
4. For the rotated images, it is apparent that the results are great.

Skew images:
1. At first, we need to know the real coordinate storing the image in MATLAB, which is a bit different from the coordinate we used often. We can find that the top-left point is the origin, the x-axis denotes the width of the image and the y-axis denotes the height of the image.
2. The angle θ in the image below, denotes the skewed angle when shearing the image along x-axis, i.e. the angle θ is used in my shear matrix.

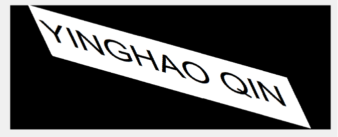


3. The implementation of skewing image is relatively easy. First, I computed the size of new image with the geometric knowledge. Then, with the forward mapping, I used the shear matrix to multiply each pixel of the original image.
4. The skewed results clearly show my result is great.

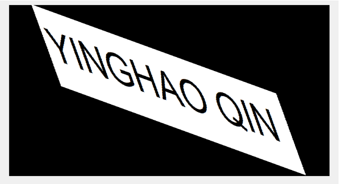
Fill the available spaces for your answers. Do not extend the spaces and do not change the formatting of the pages. Use the same font size and the same number of lines as in the original file

Question 1(b):

θ_1=20 clockwise and θ_2=50



θ_2=50 and θ_1=20 clockwise



Your comments:

From the pictures above, the two images are totally different. In other words, it is clear that the sequence of transformations will have a great impact on the final results.
In addition, the points in the lower left corner and the upper right corner of the image are close to the background edge. This is because the images are shewed along y-axis.

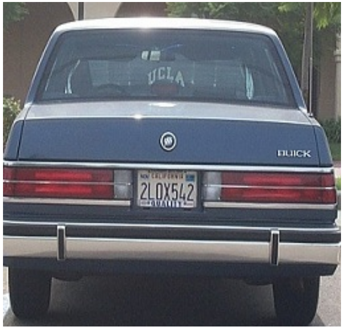
1. The implementation of the combination of rotation and skew transformations is convenient. I just need to combine the two transformation functions with different orders, which I implemented in last question.
2. By comparing the two images, it is clear that the two images are different, i.e. the order of transformations will have a big impact on the final results. And it can be explained by that the multiplication of matrix is not commutative.
3. We can find that the white areas in the both images are against the top and bottom boarders. This is due to the quality of skew operation – the height of the image is unchanged when shearing the image along x-axis.

Question 2(a):

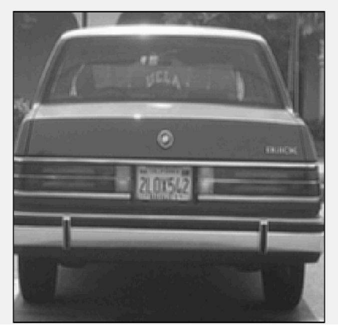
Designed kernel:

$$\frac{1}{9}*\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

Original image



Averaged image



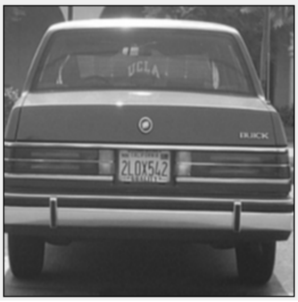
Your comments:

The averaged image seems smoother, but the license plate number in the image are more difficult to recognize. So we can get a conclusion that this method will make the image smoother, but it will also make the image lose a lot of details.

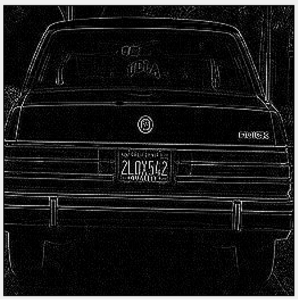
1. The steps I implemented the operations: Firstly, transfer the original image into grey image for follow-up processes. Then, normalize the kernel to avoid some pixel values exceeding the range (0 - 255). Finally, convolve the image with the kernel without border handling.
2. The averaged filter smooths the image, but it also makes the image fuzzy. For example, we can find it becomes more difficult to recognize the license plate. This is because every pixel value is affected by its neighbours i.e. each pixel value is an averaged value in the kernel size of the image.
3. There is a black border of the averaged image. In this case, it is not necessary to handle the border problem. Commonly, the image border can be handled with any padding technique (zeros, replicate or mirror).

Question 2(b):

Filtered image with kernel A



Filtered image with kernel B



Your comments:

Filtered image with kernel A is smoother because kernel A is a Gaussian filter which can eliminate noise.
Moving on to the second image, it is clear to recognize the edge of the car. In other words, kernel B is a Laplacian filter which can be used for sharp the edge. However, this image has much noise as well.

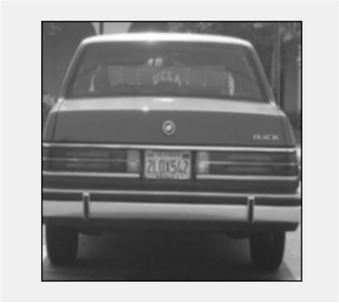
For kernel A:
1. Kernel A is a Gaussian filter. We can find that the effect of the kernel is almost similar to that of the mean kernel in Question 1 – the image gets fuzzy, but the image filtered by kernel A is clearer.
2. There are still some differences in the mean filter and Gaussian filter. Gaussian filter is used to calculate the weighted average. In detail, the highest weight will be given to the pixel corresponding to the center of the filter, and the further away from the center of the mask, the lower the weight of pixel it is.
3. According to the given kernel A and Gaussian formula $G_\sigma(k,l) = \frac{1}{2\sigma^2\pi} e^{-\frac{k^2+l^2}{2\sigma^2}}$, the sigma of kernel A is around 0.7979.

For kernel B:
1. Kernel B is a Laplacian filter, which is used to detect edges. According to the second image, it is clear to recognize the edge of the car. Besides, the given filter is sensitive to horizontal and vertical edges in images.
2. Laplacian filter makes use of the second derivative of the image. Edge points are zeros of the second derivate, which is why the sum of weights of Laplacian kernel equals to 0.

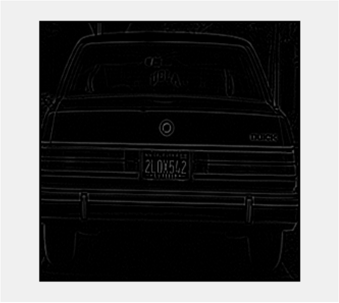
Fill the available spaces for your answers. Do not extend the spaces and do not change the formatting of the pages. Use the same font size and the same number of lines as in the original file

Question 2(c):

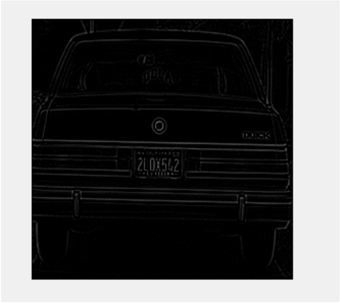
A followed by A



A followed by B



B followed by A



Your comments:

The first image shows that it is very smooth. Using Gaussian filters twice only will make the image smoother.
In the second image, it uses Gaussian filter to denoise at first and then uses Laplacian filter to sharp the edge. However, this image is too dark and hard to identify.
In the last one, it uses Laplacian filter to sharp the details and then uses Gaussian filter to denoise. This image seems lighter, but it is still hard to identify the license plate number.

For the first image:
1. Comparing with the first image in 2(b), we can find that it is fuzzier, since it did Gaussian operations twice which made the image lost more details.

For the two right images:
1. The two images are almost the same, though there are subtle differences when I compare each pixel value of their corresponding image matrices. This is because of the property of convolution – commutativity, so the order of convolution does not influence the result.
2. As for the subtle differences in the two images, this is because the conversion of data types in code results in some different values. In brief, the data type 'double' is more precise for data processing but the data type 'uint8' is used for image storage.
3. Besides, according to the commutative and associate properties of convolution, we can try to convolve kernel A and kernel B firstly, then use the gained matrix to convolve the image. However, there will be a new problem about how to cope with the border problem. Maybe we could try use the mirror method to handle it, which can keep more information.

Question 2(d):

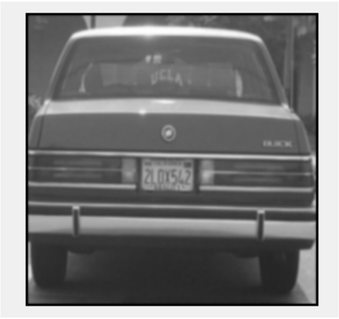
Extended kernels of A and B (5x5):

```
A=[ 0.0005    0.0049    0.0108    0.0049    0.0005;
    0.0049    0.0521    0.1142    0.0521    0.0049;
    0.0108    0.1142    0.2504    0.1142    0.0108;
    0.0049    0.0521    0.1142    0.0521    0.0049;
    0.0005    0.0049    0.0108    0.0049    0.0005;];
```

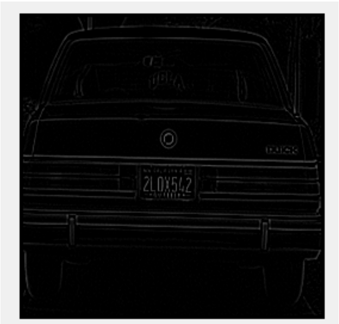
```
B = [0     0     0     0     0;
     0     0     1     0     0;
     0     1    -4     1     0;
     0     0     1     0     0;
     0     0     0     0     0;];
```

Results obtained by applying 5x5 kernel:

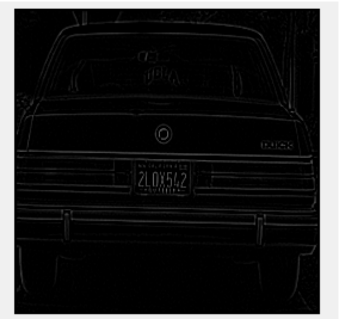
A followed by A



A followed by B



B followed by A



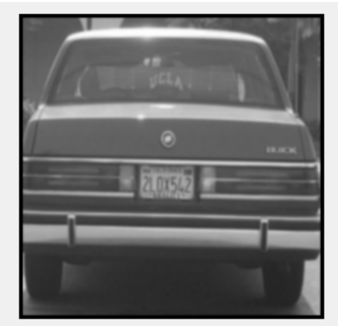
Extended kernels of A and B (7x7):

```
A7=[0.0000    0.0000    0.0001    0.0002    0.0001    0.0000    0.0000;
    0.0000    0.0005    0.0049    0.0108    0.0049    0.0005    0.0000;
    0.0001    0.0049    0.0520    0.1140    0.0520    0.0049    0.0001;
    0.0002    0.0108    0.1140    0.2500    0.1140    0.0108    0.0002;
    0.0001    0.0049    0.0520    0.1140    0.0520    0.0049    0.0001;
    0.0000    0.0005    0.0049    0.0108    0.0049    0.0005    0.0000;
    0.0000    0.0000    0.0001    0.0002    0.0001    0.0000    0.0000;];
```

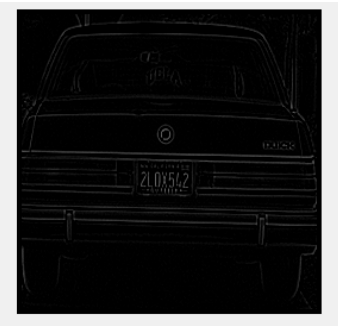
```
B7 =[0     0     0     0     0     0     0;
     0     0     0     0     0     0     0;
     0     0     0     1     0     0     0;
     0     0     1    -4     1     0     0;
     0     0     0     1     0     0     0;
     0     0     0     0     0     0     0;
     0     0     0     0     0     0     0;];
```

Results obtained by applying 7x7 kernel:

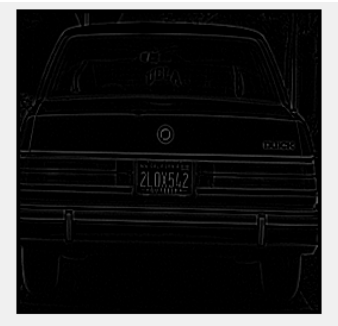
A followed by A



A followed by B



B followed by A



Your comments:

Comparing these images, it is clear that using a larger size kernel has a greater impact on the results. For example, image 6 is blurrier than image 3. This is because more pixels have an impact on the center pixel when the convolution operation occurs.
Among the six images in this case, image 2 and 5 are more likely to have actual value – it is more convenient to identify the license plate number.

For the extended kernel A and B:

1. According to the given 3×3 kernel A and Gaussian formula $G_\sigma(k,l) = \frac{1}{2\sigma^2\pi} e^{-\frac{k^2+l^2}{2\sigma^2}}$, the sigma of kernel A is around 0.7979. Using the formula again with the sigma as 0.7979, we can get larger size Gaussian kernels which has the same effect of the given 3×3 kernel A.
2. In order to obtain exactly the same effect of the 3×3 kernel B, kernel B should be padded zeros along the borders of the kernel according to the desired kernel size. This is commented in assignment feedback by Alessio (our lab assistant). Additionally, I designed other kinds of Laplacian kernel with any size, the results obtained by applying them are clearer.

For the results obtained by applying extended kernels:
1. Compared to the left image in 2(c), the results obtained by applying extended kernel A seem to have no obvious change. But we can find that the black border is becoming wide with the increasing of the kernel size. This is because I do not handle the border problem i.e. the border is not filter.
2. For all the results obtained by applying A and B, we can find the extended kernels have the same effect on the filtered image with the given 3×3 kernels.

Question 3(a):

Two non-consecutive frames:

Image 1

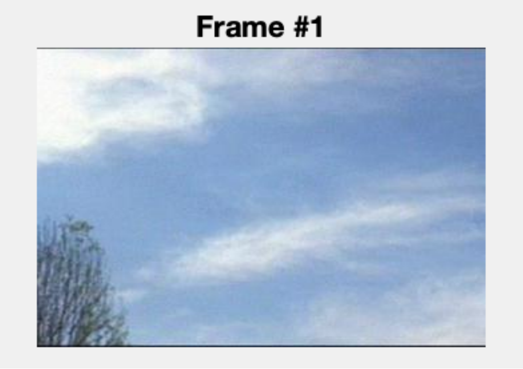
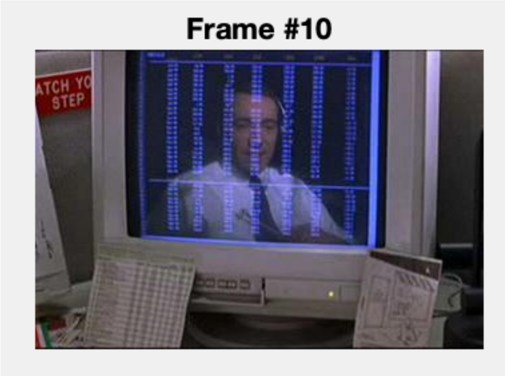
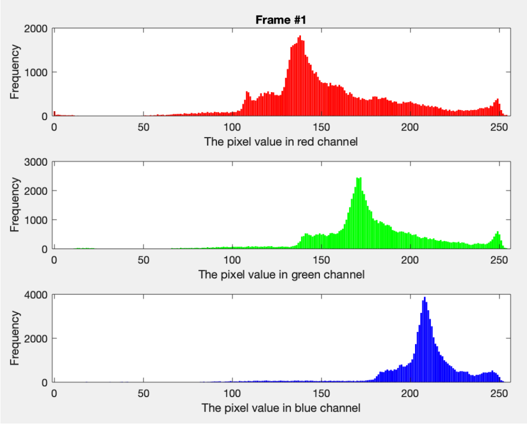


Image 2

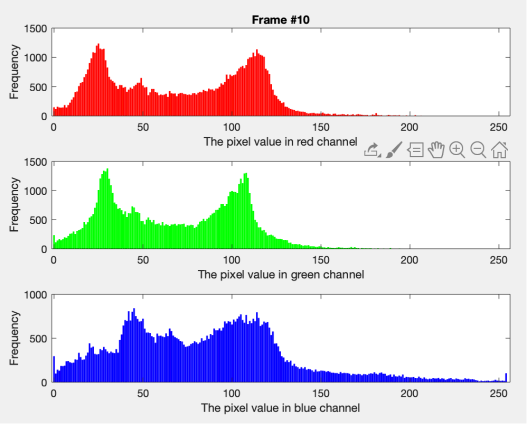


Corresponding colour histograms:

Histogram 1



Histogram 2



Your comments:

There is a big difference in color between the first frame and the 11th frame. As a result, their respective color histograms are quite different as well. For example, the peak values of the left histograms are on the right, and the peak values of the right histograms are on the left. This is probably because the right image is darker than the left one and we know a true that 0 denotes black and 255 denotes white in this case.

In addition, gray levels are also different with respect to the histogram. It is amazing that the combination of gray levels causes colorful images.

1. How to get the color histogram? It is an algorithm that maps the quantization levels into the frequency of each quantization level in the image i.e. the data in the color histogram is obtained by counting how many times each pixel value occurs for the corresponding color channel in the image.

2. Histogram 1 shows the distributions of different pixel values in different channel could be different. However, histogram 2 shows the distributions of different pixel values in different channel could be similar as well.

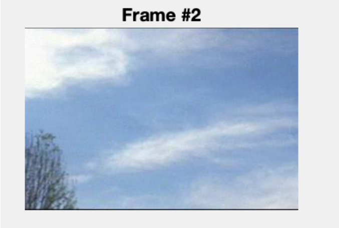
3. From histogram 2, we can find a couple of peaks in each channel, which is because image 2 can be divided into several parts based on the distribution of color i.e. the color partition in image 2 is more obvious.

4. According to histogram, we can recognize those pixel values with the high frequency which mainly represent the image. And for pixel values with a very low frequency, they could be regarded as noise in a sense.

Question 3(b):

Example 1:

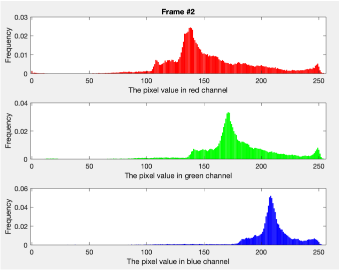
I_t



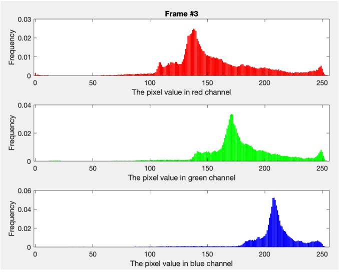
I_{t+1}

Histograms:

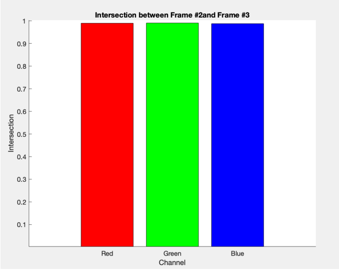
Histogram of I_t



Histogram of I_{t+1}



Intersection result



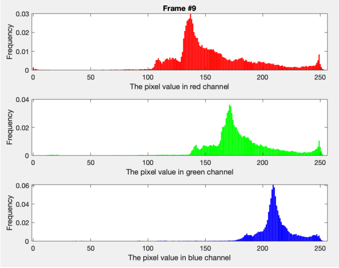
Example 2:

I_t

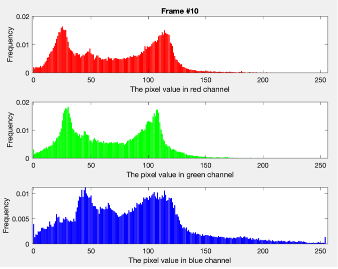
I_{t+1}

Histograms:

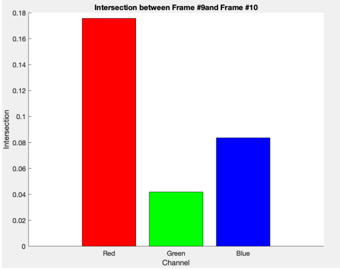
Histogram of I_t



Histogram of I_{t+1}



Intersection result



Your Comments:

$$\boldsymbol{min_j(I_t, I_{t+1}) \Big/ \sum_{j=1}^{n} min(I_t, I_{t+1})}$$

Where j ranges over each colour in the histograms.
I use the equation above to normalize the intersection result. As shown in the figure above, the label of the y axis is the frequency of the pixel numbers.
According to the two examples, it is clear that the two intersection results are different, which can be used to classify the images i.e. the similar consecutive images have the similar intersection results.

1. The colour histograms shown in 3(b) are different from those in 3(a). Here the histograms are normalized by the formula below.

$$\boldsymbol{H_c(i) = h_c(i) / \sum_{i=0}^{255} h_c(i)}$$

Here, c denotes the channel, $h_c(i)$ denotes the original frequency for pixel quantization value i in the corresponding channel, $H_c(i)$ refers to the normalized value for quantization value i in the corresponding channel. So $H_c(i)$ has the properties, $0 \le H_c(i) \le 1$ and $\sum_0^{255} H_c(i) = 1$.

2. By comparing the normalized and non-normalized histograms, we can find that normalization will not change the shape of histogram. After normalization, the label of Y axis denotes the weighted value of certain pixel value in a channel of the image.

3. Similarly, normalization will not change the shape of intersection histogram. Without normalization, the label of Y axis represents the number of pixels which are overlapping in the two corresponding frames. With normalization, the label of Y axis represents the similarity of the two frames.

4. For example 1 in 3(b), the intersection values in all the three channels are close to 1, which means that there is a high similarity between frame 2 and frame 3. In contrast, for example 2 in 3(b), the intersection values in all the three channels are obviously lower than those in example 1 (lower than 0.18), meaning that there is a big difference between frame 9 and frame 10.

5. By applying the normalization method, for two highly related histograms (images), the intersection value will be close to 1 at each channel; for two highly different histograms (images), the intersection value will be closer to 0 at each channel. In brief, the higher the intersection value is, the more similar the two images in terms of pixel value are.

Fill the available spaces for your answers. Do not extend the spaces and do not change the formatting of the pages. Use the same font size and the same number of lines as in the original file

Question 3(c):

Comments:
The intersection value represents the overlap of the corresponding two histograms for a given video.

I cannot use it to make a decision where the scene in the video changes. This is because this image has only statistics, but there is no position information for each pixel.

The histogram intersection technique is not obvious for changes. For example, the intersection results are similar to the histograms. This technique does not obviously show the changes between histograms.

Histogram calculation can be used for brightness purposes. Not only in brightness, but histograms are also used in adjusting contrast of an image. Another important use of histogram is to equalize an image. And last but not the least, histogram has wide use in thresholding. This is mostly used in computer vision.
Histogram intersection can be used for image classification.

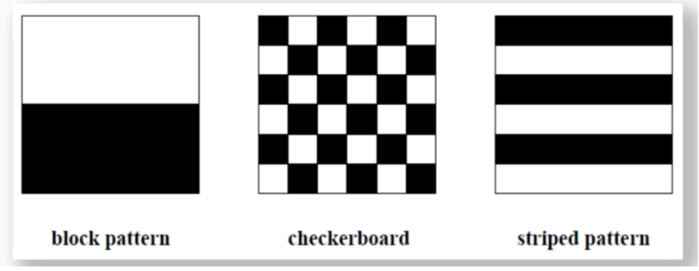
1. The intersection value represents the similarity of two consecutive frames for a given input video.

2. For two similar scenes in a video, the intersection value is close to 1 and the amplitude of variation of consecutive intersection values is small when playing the video, whereas when the scene changes in a video, the intersection value will drop dramatically, even close to zero. Consequently, we can set up a threshold of intersection values change to make a decision about whether the scene in the video changes.

3. This technique is robust to certain geometric transformations such as rotation, scaling, mirroring, shear etc., but not robust to texture etc.

4. Histogram intersection could fail sometimes. For instant when coping with the images below, the histograms of the three images are the same. In this case, if the three frames are consecutive in a video, the change among the three frames are not identified by observing the intersection histogram.

same intensity distribution (50% black & 50% white pixels)
three different textures



5. Application of histogram: quick indication as to whether or not the image covers the whole brightness range of digitizer; tool for image segmentation, e.g. separating objects from background.
Application of histogram intersection: comparing the similarity of a picture with a set of pictures in the dataset to find the most similar one.