Amirkabir University of Technology
(Tehran Polytechnic)

Assignment 2
When Cameras Fail: Image Enhancement in Spatial Domain

Homeworks Guidelines and Policies

- ***What you must hand in.*** It is expected that the students submit an assignment report (HW2_[student_id].pdf) as well as required source codes (.m or .py) into an archive file (HW2_[student_id].zip).
- ***Pay attention to problem types.*** Some problems are required to be solved *by hand* (shown by the icon), and some need to be implemented (shown by the icon).
  Please don't use implementation tools when it is asked to solve the problem by hand, otherwise you'll be penalized and lose some points.
- ***Don't bother typing!*** You are free to solve by-hand problems on a paper and include picture of them in your report. Here, cleanness and readability are of high importance. Images should also have appropriate quality.
- ***Reports are critical.*** Your work will be evaluated mostly by the quality of your report. Don't forget to explain what you have done, and provide enough discussions when it's needed.
- ***Appearance matters!*** In each homework, 5 points (out of a possible 100) belongs to compactness, expressiveness and neatness of your report and codes.
- ***Python is also allowable.*** By default, we assume you implement your codes in MATLAB. If you're using Python, you have to use equivalent functions when it is asked to use specific MATLAB functions.
- ***Be neat and tidy!*** Your codes must be separated for each question, and for each part. For example, you have to create a separate .m file for part b. of question 3. Please name it like p3b.m.
- ***Use bonus points to improve your score.*** Problems with bonus points are marked by the icon. These problems usually include uncovered related topics or those that are only mentioned briefly in the class.
- ***Moodle access is essential.*** Make sure you have access to Moodle because that's where all assignments as well as course announcements are posted on. Homework submissions are also done through Moodle.

- ***Assignment Deadline.*** Please submit your work **before the end of April 30th**.
- ***Delay policy.*** During the semester, students are given 7 free late days which they can use them in their own ways. Afterwards there will be a 25% penalty for every late day, and no more than three late days will be accepted.
- ***Collaboration policy.*** We encourage students to work together, share their findings and utilize all the resources available. However you are not allowed to share codes/answers or use works from the past semesters. Violators will receive a zero for that particular problem.
- ***Any questions?*** If there is any question, please don't hesitate to contact me through **ali.the.special@gmail.com**. You may also find me in the pattern recognition and image processing lab, 3rd floor, CEIT building.

1. Mathematics of Image Point Processing Operations (8 Pts.)

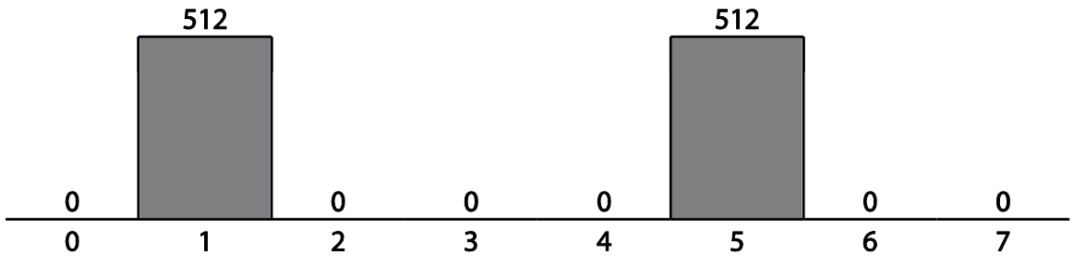
Keywords: *Image Point Processing, Image Histogram, Histogram Equalization, Histogram Matching (Specification), Image Negative, Bit-plane Slicing, Image Thresholding, Contrast Stretching*

The first task contains several problems in the topic of **Point Processing**. Point processing operations are an important group of **Image Enhancement** techniques in **Spatial Domain**, which focus on enhancing images solely with manipulating pixel intensity values. The problems here are aimed to make sure you've theoretically mastered those operations.

First, consider the following 3-bit image of the size 8×8:

2	3	2	2	3	2	2	1
3	3	5	3	5	3	2	2
3	5	6	5	6	3	2	2
3	6	6	6	6	5	3	3
3	6	7	7	6	6	5	3
3	6	6	6	6	6	3	3
3	5	6	4	6	6	5	3
2	3	3	3	4	6	4	2

a. Find the digital negative of the image.
b. Perform bit-plane slicing and write down all bit-planes representations of the image.
c. Compute the image histogram.
d. Determine a "valley" in the histogram, and threshold the image at this value. Write down the resultant image matrix.
e. Apply linear scaling for stretching the gray levels of the image to the range of 0-255. Write down the result matrix.
f. Apply histogram equalization to the image. Write the resultant matrix as well as the equalized histogram.
g. Suppose you want to shape the histogram of the above image to match that of a second image, given below:



Use histogram specification to devise a gray-level transformation that shapes the histogram of the original image to look like the second. Note that you have to adopt a strategy to handle certain cases, including when two input values map to the same output value, and no input values map onto a particular output value.

Now, assume an image with the following gray-level histogram:

$$p_r = \frac{e^r - 1}{e - 2}, \quad r \in [0,1]$$

h. Find a gray-level transformation $s = T(r)$ to make the transformed histogram p_s uniform, i.e.:

$$p_s = 1 \quad s \in [0,1]$$

Note: Make sure to show all intermediate steps in the calculations.

2. Practicing Image (De)Convolution Calculation (12 Pts.)

Keywords: *Image Spatial Filtering, Kernel (Mask), Image Convolution, Image Smoothing, Image Sharpening, Image Gradient*

Image Filtering is another way to enhance images in spatial domain. It is done through convolving a predefined **Kernel** over an image. So before jumping into the world of pixels and colors, it's good to practice some **2D Convolution** problems first.

0	7	7	0	0	3	3	3
0	7	0	0	0	3	7	3
0	7	3	3	3	3	3	3
0	7	0	0	0	7	0	0
0	0	7	0	3	3	3	0
3	0	7	0	3	3	3	0
7	0	7	0	3	3	3	3
3	3	0	0	0	7	0	7

First, a 3-bit image of the size 8×8 is given. Convolve each one of the given kernels (a) to (h) with this image and write the output image matrix. Also describe how each mask affects the input image.

-1	-1	-1
-1	8	-1
-1	-1	-1

(a)

-1	0	1
-1	0	1
-1	0	1

(b)

-1	-1	0
-1	0	1
0	1	1

(c)

0	-1	0
-1	4	-1
0	-1	0

(d)

1	1	1
1	1	1
1	1	1

(e)

-1	-1	-1
2	2	2
-1	-1	-1

(f)

0	-1	-1
1	0	-1
1	1	0

(g)

-1	2	-1
-1	2	-1
-1	2	-1

(h)

Now, let's do something more exciting. We're going to do the reverse process, i.e. **Image Deconvolution**. It's a technique widely used to restore the original image by estimating the convolution kernel. Here, you are asked to specify each one of the 3×3 kernel which was convolved with the original image to yield the outputs (i) to (p). Please clearly explain the strategy you used for each part.

Note: Rounded intensities are shown by blue, and out-of-bound values are shown by red.

2	2	2	1	1	2	2	2
2	3	4	2	2	3	3	2
2	3	3	1	2	3	3	2
2	3	3	2	2	3	2	1
1	3	2	2	2	3	2	1
1	3	2	3	2	3	2	1
2	3	2	2	2	3	3	2
1	2	1	1	1	2	3	1

(i)

7	7	7	7	3	6	7	6
7	7	7	0	3	7	7	7
7	7	7	7	7	7	7	6
7	7	7	0	7	7	7	0
0	7	7	7	6	7	6	3
3	7	7	7	6	7	6	3
7	7	7	7	6	7	7	6
6	6	3	0	7	7	7	7

(j)

0	5	2	0	0	2	3	2
0	7	3	1	1	3	4	3
0	7	1	1	1	4	3	2
0	5	3	1	2	4	2	1
1	2	5	0	2	4	2	0
3	0	7	0	3	3	3	1
4	1	5	0	2	4	2	3
3	1	2	0	1	3	1	3

(k)

0	7	7	0	0	7	2	7
0	7	0	0	0	2	7	2
0	7	5	7	7	0	2	7
0	7	0	0	0	7	0	0
0	0	7	0	7	0	7	0
7	0	7	0	6	3	6	0
7	0	7	0	7	0	6	5
5	7	0	0	0	7	0	7

(l)

7	0	0	7	6	0	0	0
7	0	7	7	7	0	0	0
7	0	0	0	0	0	2	0
7	0	7	7	7	0	7	7
7	7	0	7	0	0	0	6
0	7	0	7	0	0	0	7
0	7	0	7	0	0	2	0
0	0	7	7	7	0	7	0

(m)

1	4	3	1	1	2	3	2
2	4	3	1	1	3	4	3
2	4	3	2	2	3	3	2
1	3	3	1	2	3	2	1
1	2	3	2	2	3	2	1
2	3	4	3	2	3	2	1
3	3	3	2	2	3	3	2
2	2	1	1	1	3	3	2

(n)

0	0	0	0	0	0	0	0
0	4	5	0	0	0	0	0
0	0	0	0	0	0	7	7
7	7	2	2	3	0	3	6
1	4	0	0	0	2	0	0
0	0	0	0	0	0	0	0
0	1	7	7	2	0	0	0
7	7	7	7	7	7	7	7

(o)

2	0	0	0	1	0	0	0
4	0	4	3	2	0	0	0
4	0	1	0	0	0	0	0
2	0	3	4	2	0	2	2
2	3	0	3	0	0	0	1
0	5	0	5	0	0	0	2
0	3	0	3	0	0	1	0
0	1	0	3	1	0	2	0

(p)

3. Image Enhancement Techniques in Action (16 Pts.)

Keywords: *Image Enhancement, Image Filtering, Image Point Processing*

Image Enhancement techniques are used in pretty much every area that deals with images. According to their needs, experts in these areas use these techniques to better visualise images and analyse their contents in a more detailed way.

Here, you are introduced with four different applications – some realistic and some fictional. Images taken in each one of these applications suffer from one or several image degradations, e.g. low-contrast. Your goal is to first identify these degradations, and then try to reduce them by using techniques you learned in the area of image enhancement so far.

Note 1: Feel free to use as many methods as you like, but please implement them yourself.

Note 2: The outputs of the color images must be also a color image.

I. Up close with the enemy: fighting against Coronavirus

You hate Coronavirus? Well, do something! A team of medical experts gave you some microscopic images of Covid-19 and asked you to enhance them, so that they could study them in more details. See what you can do for them.

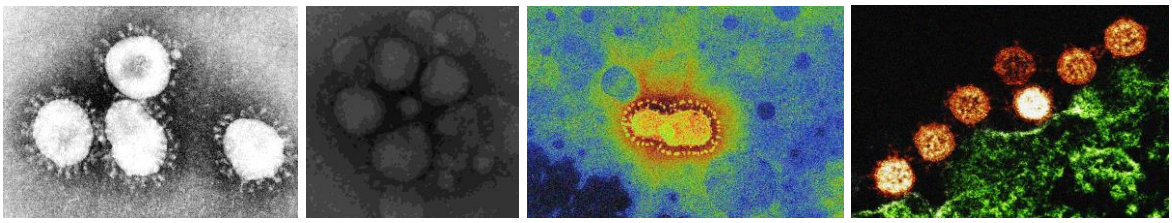


Figure 1 Several microscopic images of the Covid-19 virus in presence of different image degradations

II. Barcode scanner system

Apart from image enhancement, one important application of image filtering is feature extraction. Assume you are asked to implement a barcode scanner system, capable of decoding all types of barcodes, such as QR code, MaxiCode and Aztec Code. Your first step is to extract meaningful features, like vertical lines, dots, etc.

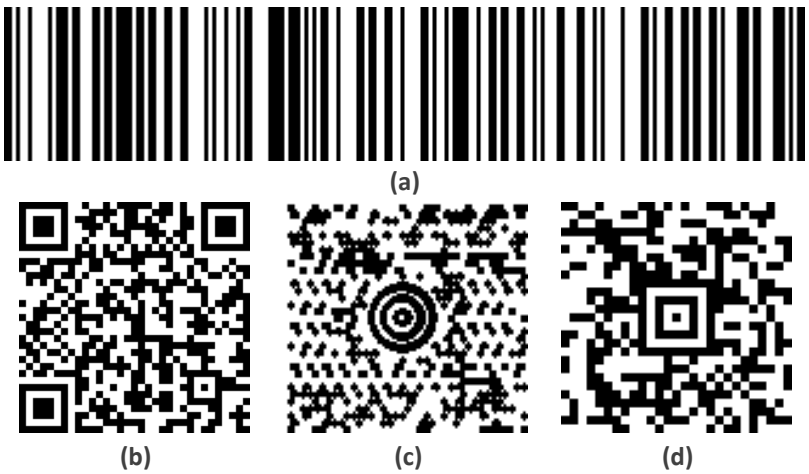


Figure 2 Different types of barcodes with different features (a) Classic barcode (b) QR Code (c) MaxiCode (d) Aztec Code

III. NASA needs help

Those magnificent images of the night sky you see once in a while are much different without image enhancement techniques. Due to limitations of photography instruments and severe photography conditions, these images suffer from various degradations such as noise and motion blur. Let's deal with some unedited night sky images and see how we can handle them.

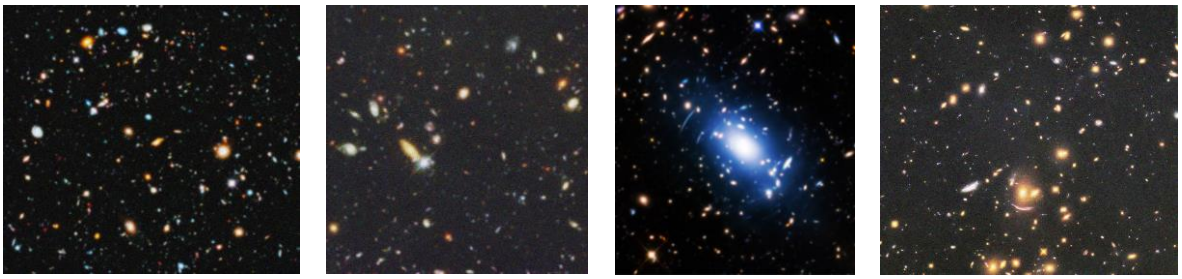


Figure 3 Because of instruments limitations and long-distance photography, images of the night sky usually suffer from noise and are often blurry

IV. Ghost-hunting

There are some images which are claimed to be taken from real ghosts. A ghost-hunting team needs your assistance in order to investigate whether these images are real or fake. The goal is to enhance these images, so that the ghost-hunter experts have an easier task to verify them. Try to enhance the images so that the ghosts become clearly visible in white.

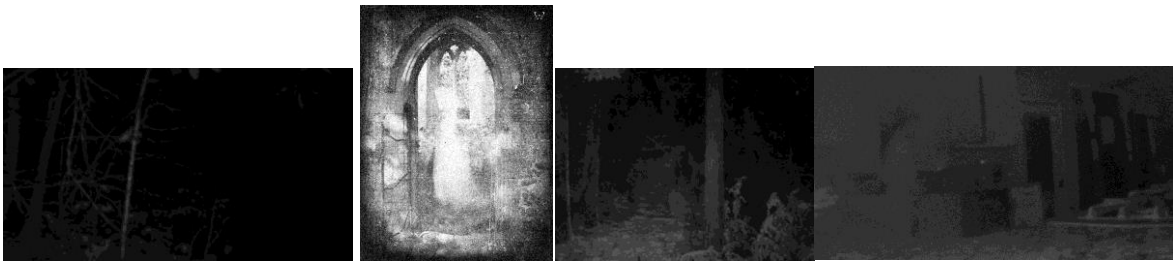


Figure 4 Images of the ghosts are taken in uncontrolled conditions and therefore suffer from low contrast and brightness

4. Motion Detection Algorithm Based on Bit-plane Slicing (10 Pts.)

Keywords: *Motion Detection, Bit-plane Slicing, Image Difference*

If you think **Bit-plane Slicing** of an image is an old-fashioned image processing operation with no use, think again. Because it not only comes to use in applications like **Image Compression**, but can even handle some high-level machine vision tasks, namely **Motion Detection**. Let's see how.

Imagine two consecutive frames are extracted from a video sequence, and the goal is to detect the moving object(s) inside these frames. One may quickly think of **Image Difference**, but there are more efficient ways available. Assume we first convert the frames into 8-bit grayscale images and then slice grayscale pixel values into eight constituting bits using:

$$a_k = \left\lfloor \frac{I}{2^k} \right\rfloor \bmod 2$$

where I is a grayscale pixel value, k is the bit number and a_k is the bit value of the corresponding bit number (Figure 5).

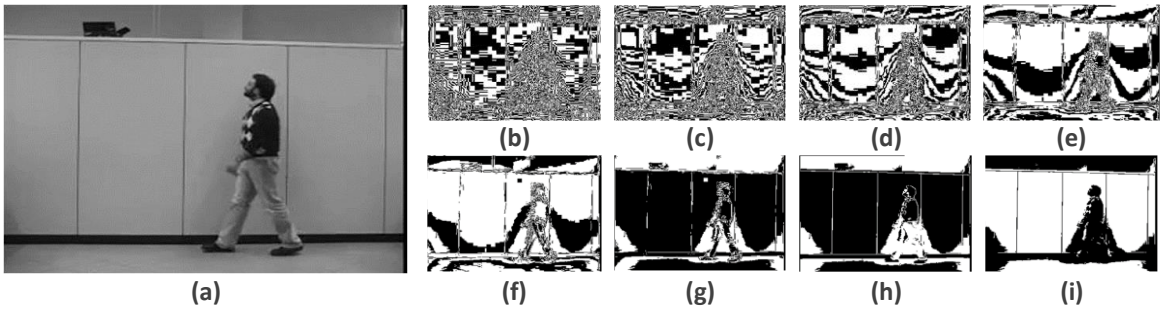


Figure 5 Slicing a frame into bit-plane representations (a) Original image (b) to (i) Bit-plane representations from least to most significant bits

Then, it would be easily possible to compare consecutive frames using XOR function of the corresponding bit-planes, i.e.:

$$c_k = a_k \oplus b_k$$

where a_k, b_k and c_k are bit values of frame 1 pixel, frame 2 pixel and resultant bit value respectively (Figure 6).

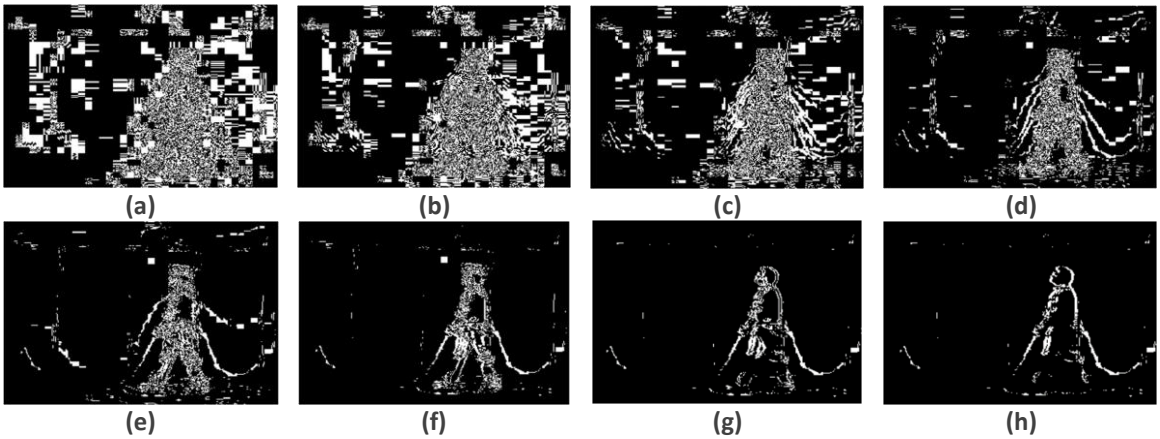


Figure 6 The results of performing XOR function on the corresponding bit-plane representations of the two frames (a) Least significant bit (b) to (g) intermediate bit-planes (h) Most significant bit

Finally, a grayscale image can be obtained by merging higher bit-planes of the resultant of the XOR operations:

$$Y = \sum_{k=4}^{7} 2^k \times c_k$$

This image is expected to highlight moving regions of the two frames. Some post-processing operations may also be needed to get a more clear result (Figure 7).

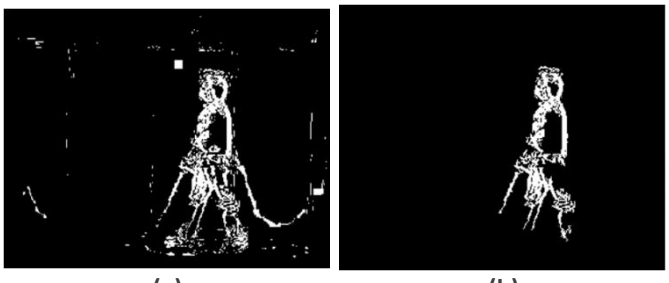


Figure 7 Reconstructed image using the four highest bit-planes (a) Before filtering (b) After filtering

Now, let's do this ourselves.

a. Implement a function `bitplane_slice()` which takes an input image, converts it into grayscale and then slices it into its bitplanes. Perform bit-plane slicing on the given two frames using your function, and display the results.
b. Perform XOR function on the corresponding bit-planes of the two frames, and display the results.
c. Use the 4 highest bit-planes to obtain moving parts of the two frames.
d. Apply proper image enhancement techniques to obtain a final clean image.

(a) **(b)**

Figure 8 Inputs of this part contain two consecutive frames taken with a stationary camera, with a man walking (a) Frame 1 (b) Frame 2

5. Become an Artist Through Image Filtering (12 Pts.)

Keywords: *Digital Art, Image Filtering, Image Thresholding*

Ever wanted to be an artist? Did you fancy being a painter and eventually end up in the field of artificial intelligence? Well, it's not late yet. Because **Image Filtering** techniques makes it possible to create amazing painted-like images no one believes were created in a few lines of code.

I. Image Sketch.

The first objective concerns pencil sketch. A simple algorithm for converting a color image into its sketch peer is given (Figure 9). Apply these steps on the input images provided for this problem.

(a) **(b)** **(c)** **(d)** **(e)**

Figure 9 The process of converting an image into a sketch using the proposed algorithm (a) Original image (b) Converting the image into grayscale (c) Inverting the resultant grayscale image (d) Smoothing the negative image using an appropriate kernel (e) Dividing the blurred image by the grayscale image and obtaining the final sketch

a. **Grayscale conversion.** Convert the input image to grayscale using the following formula, and give the result a name like `gray_img`:

$$Y = 0.299R + 0.587G + 0.114B$$

b. **Finding image negative.** Subtract pixel values from 255 and obtain the image negative.
c. **Blurring the image.** Use a smoothing filter to blur the image, and give it a name like `blurred_img`.
d. **Color dodging.** Divide `blurred_img` by `gray_img` to lighten the image and get a sketch. You must do `result=blurred_img*255/(255-gray_img)`.

e. Investigate the effect of smoothing kernel size on the result.

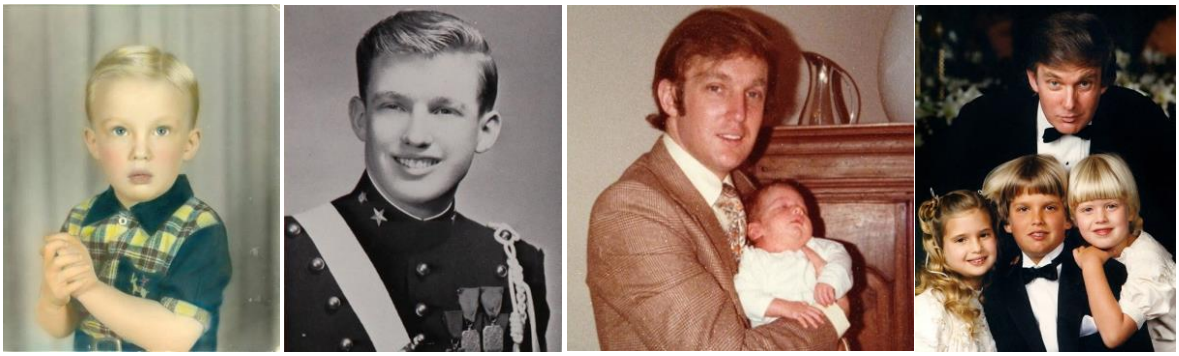


Figure 10 Input images that are asked to turn into a sketch

II. Oil Painting.

Now comes the second task, which is rather more complicated. The goal here is to add art effects to the input image and create an oil painting. Load the input image and apply the following image manipulation operations to it.

(a) **(b)**

Figure 11 Example of converting an image into an oil painting equivalent (a) Original image (b) Oil painting image

f. **Grayscale conversion.** Like the previous task, first convert the input image to grayscale using the same formula.

g. **Perform a convolution operation.** Define a mask around the current pixel and apply it to each pixel of the image. It's up to you to find an appropriate mask type as well as its size.

Figure 12 You are required to convert this image so that the result looks like an oil painting image

a. **Finding the most frequent local pixels.** For each of the masked area, compute the intensity histogram and based on that pick the most frequent pixel value, and replace the current pixel with this value.

b. **Update intensity values.** Make a copy image G of the original color image, and for each pixel (x, y) in the result image I, in the local neighborhood defined by the mask in the previous part:
   - Find all pixels which have the same value of $I(x, y)$
   - Get the intensities of those pixels in image G
   - Calculate the average intensities of those pixels in each band
   - Replace $G(x, y)$ with the average value in each corresponding band

c. Use any image enhancement techniques – if necessary – to improve or diversify the visual effect of the results.

d. Try different masks with different sizes and compare the results.

Note: Make sure to display and save the intermediate results as well as including them in your report.

6. Protect the Law: Traffic Monitoring by Image Averaging and Thresholding (16 Pts.)

Keywords: *Background Subtraction, Foreground Detection, Object Detection, Image Averaging, Image Thresholding*

Foreground Detection, also known as **Background Subtraction**, is a machine vision task in which the goal is to detect foreground objects in a given image. These techniques usually analyse video frames recorded with a stationary camera, and distinguish foreground from background based on the changes taking place in the foreground. They are widely used in various applications, such as traffic monitoring, object tracking, human action recognition, and so on.

(a) **(b)**

Figure 13 After detecting background image, it's easy to subtract the original image from the obtained background and find foreground image (a) Original image (b) Background image

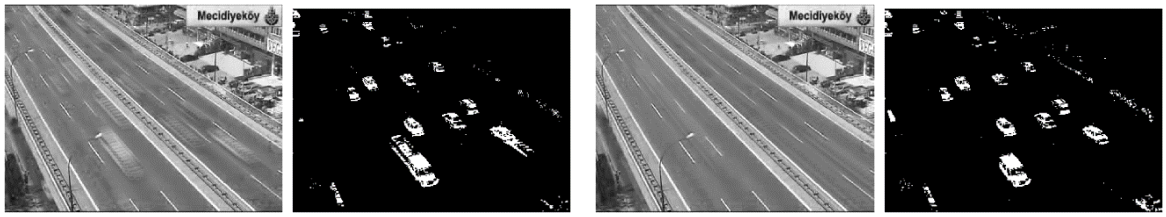
As you learned in the class, **Image Averaging** is a technique used to reduce noises if multiple images of the same scene are available and noise has zero mean and be uncorrelated. In another word, assuming n different noisy images I_i, then

$$\bar{I}(x,y) = \frac{1}{n}\sum_{i=1}^{n} I_i(x,y)$$

can be shown to be less noisy.

Now based on this framework and applying **Image Thresholding**, we can introduce a simple background detection algorithm in a video sequence, where background is the mean of the previous n frames F_i:

$$B(x,y) = \frac{1}{n}\sum_{i=1}^{n} F_i(x,y)$$



$n=10$ $n=50$

Figure 14 The number of images used in the averaging process considerably affects the "cleanness" of the background obtained by the algorithm, and therefore the output foreground image

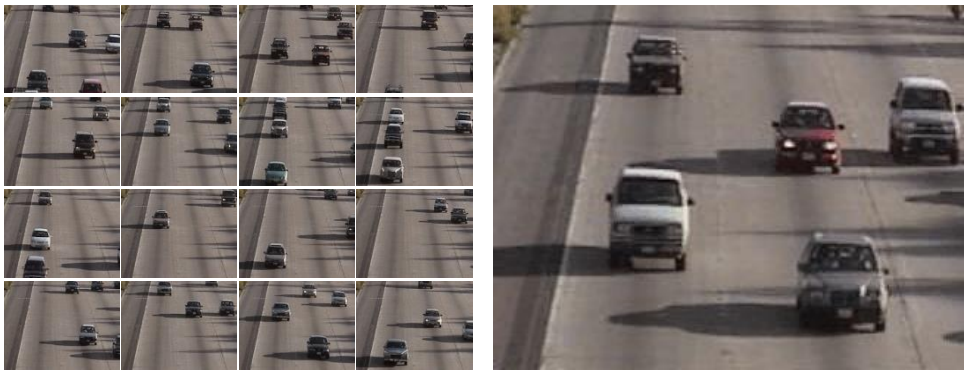
Similarly, if the background is more likely to appear in a scene, we can use the median of the previous n frames as the background model:

$$B(x,y) = median\left[F_i(x,y)\right]_{i=1}^{n}$$

Here, your objective is similar. You are provided with 80 frames of a 5 seconds video sequence recorded from a crowded highway, and another test frame of the same scene in a different time. The goal is to detect all cars in the test image by first finding a car-free image of the highway (background) and then subtracting the test image from it.

a. Estimate the background image using the first n frames. Set $n = 5, 10, 20, 40, 80$.
b. Subtract the test frame from the background images you obtained in the previous part.
c. Define appropriate thresholds and find foreground masks corresponding to each of the background images.
d. Use the resultant masks to extract cars in the test frame. Assign black color (0,0,0) to the remaining regions of the image.
e. Repeat this process using median operation, and compare the results.

Note: Make sure to display and save the intermediate results as well as including them in your report.



(a) **(b)**

Figure 15 Input video sequence (a) Some extracted frames (b) Test frame

7. Image *Obamafication* (!) Using Bilateral Filtering (16 Pts.)

Keywords: *Image Spatial Filtering, Gaussian Filtering, Bilateral Filtering, Image Thresholding*

Barack Obama 2008 presidential campaign was mostly represented by an iconic poster, widely known as "Barack Obama hope poster", with a stencilled face of Obama in solid red, beige and blue (light and dark), and the caption "Hope" below (Figure 16). The poster became one of the most widely recognised symbols of Obama's campaign message, spawning many variations and imitations.

Figure 16 Barack Obama famous "Hope" poster, used in his 2008 presidential campaign

The poster was created by graphic designer and street artist Shepard Fairey in just one day. In this problem, your task is to create similar effect on two given images. Through this, you'll practice a modified and improved version of **Gaussian Filtering**.

The method we introduce here combines the effects of **Bilateral Filtering** with **Image Thresholding** technique. It first applies bilateral filtering to the image, then converts the result to grayscale and maps the four specific colors to different regions of the images based on user-defined threshold values. Figure 17 depicts this procedure.

In the bilateral filter, the output pixel value depends on a weighted combination of neighboring pixel values:

$$I_F(i,j) = \frac{\sum_{k,l} I(k,l)\omega(i,j,k,l)}{\sum_{k,l} \omega(i,j,k,l)}$$

where (i, j) and (k,l) are the position of the current pixel and neighboring pixel respectively, I_F is the filtered version of the input image I, and ω is the **Bilateral Weight Function** defined as below:

$$\omega(i, j, k, l) = \exp\left(-\frac{(i-k)^2 + (j-l)^2}{2\sigma_d^2} - \frac{\|I(i, j) - I(k,l)\|^2}{2\sigma_r^2} \right)$$

where σ_d and σ_r are **Smoothing Parameters**. Increasing spatial parameter σ_d smooths larger features, while by increasing range parameter σ_r the filter becomes closer to the Gaussian filter.

(a) **(b)** **(c)** **(d)**

Figure 17 Procedure of the proposed method (a) Original image (b) The output of performing bilateral filter on the input image (c) Conversion the filtered image into grayscale (d) Using image thresholding to color the image

Now let's roll! Load the input images and use them in the following steps.

a. Implement the function `bilateral_filter()` based on the definition above.
b. Play around with the parameters and display four distinct results under arbitrary values set for these parameters.
c. Convert the image to grayscale and map the colors in table 1 to pixels based on intensity, using appropriate threshold values. You are free to color each region with your desired color.
d. Use the provided masks to separate background and foreground. Color half of the background with the color "desaturated cyan" and the other half with "lava".
e. Argue on the importance of the threshold value.

Table 1 Hope poster colors and their details

Color	Name	RGB Values
	Prussian Blue	(0, 48, 80)
	Desaturated Cyan	(112, 150, 160)
	Peach-Yellow	(250, 227, 173)
	Lava	(218, 20, 21)

(a) **(b)**

Figure 18 Input images and their masks, which are needed to separate background and foreground (a) Input 1 and its corresponding mask (b) Input 2 and its corresponding mask

8. Further Study: Guided Image Filtering (20 Pts.)

Keywords: *Image Smoothing, Edge-preserving Filtering, Guided Image Filtering*

Although **Image Smoothing** is a popular technique to reduce noise and unwanted details in images, classical smoothing methods such as **Gaussian Smoothing** also affect fine image details. There exists many extensions to these methods that try to overcome this issue, called **Edge-preserving Filtering** methods. These methods focus on smoothing images while preserving as much detail as possible.

You got familiar with one of these methods in the previous problem, i.e. **Bilateral Filtering**. While very effective, bilateral filter suffers from two shortcomings; **Staircase Effect** and **Gradient Reversal**. The first concerns intensity plateaus that causes the images appear like cartoons (although this feature came to use in the previous problem) and the second lead to introduce false edges in the image.

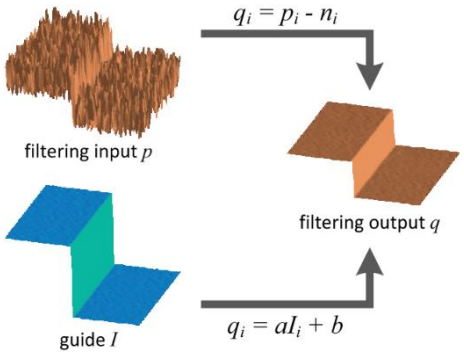


Figure 19 In guided image filtering, filtering output is a linear transformation of the guidance image. Here, n_i is the unwanted details of the input image.

A more recent approach to this problem is called **Guided Image Filtering**, which is presented in a 2013 paper with the same name, published in the IEEE Transactions on Pattern Analysis and Machine Intelligence. In this method, filtering is conducted by considering the content of another image. The output of the filtering process is a linear transform of this guidance image. While it has good edge-preserving properties, it does not suffer from bilateral filtering artefacts mentioned above.

Please read the paper attached to this homework, and answer the following questions according to your understanding of the paper:

a. Explain your understanding of the guided image filtering framework. Your explanations must cover the method procedure as well as the way it helps edge-preserving smoothing.
b. How does guided image filtering deal with gradient reversal problem?
c. In the proposed method, is it possible to use a color guidance image? Explain.
d. What are the parameters of the guided image filtering, and what is the intuitive meaning behind them?
e. Write a brief review of the paper based on your understanding of the proposed method.

Note 1: You are free to utilize relations and images from the paper, but a mere translation wouldn't be of much worth.

Note 2: Your answers may not be totally accurate, but your efforts are worthwhile.

9. Some Explanatory Questions (5 Pts.)

Please answer the following questions as clear as possible:

a. Why the discrete histogram equalisation doesn't yield a flat histogram in general? Explain.
b. Does repeatedly applying a 3×3 smoothing filter to an image converge to a certain state?
c. What is the effect of setting the LSB bit-plane to zero on the image histogram? What about setting the MSB bit-plane to zero?
d. Why a Gaussian filter in usually used before applying a Laplacian filter?
e. What will happen if instead of first smoothing the image and then computing the image gradient, we compute the image gradient first and then apply image smoothing on the resultant image?

Good Luck!
Ali Abbasi