Deep Learning for Computer Vision
Professor Vineeth N Balasubramanian
Department of Computer Science and Engineering
Indian Institute of Technology Hyderabad
Image Representation

In last lecture we spoke about image formation and now we will move on to how do you represent an image so you can process it using transformations
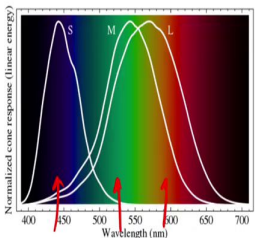
(Refer Slide Time: 00:30)

So, we did leave one question during the last lecture which is if the visible lights spectrum is VIBGYOR from violet to red, why do we use an RGB colour representation, hope you all had a chance to think about it, read about it and figure out the answer. The answer is the human eye is made up of rods and cones.

The rods are responsible for detecting the intensity in the world around us and the cones are responsible for capturing the colours and it happens that the human eye there are mainly three kinds of cones and these cones has specific sensitivities and the sensitivities of these cones are at specific wavelengths which are represented by S, M and L on this particular figure.

So, if you look at where these cones peak at that happens to be close to red, green and blue and that is the reason for representing images as red, green and blue in all honesty the peaking does not happen exactly red, green and blue, it actually happens in off colours in between but for convenience we just use R, G and B.

Some interesting facts here it happens that the M and L wavelengths here are stronger on the X chromosome, so which means males who have the XY chromosome, females have the XX are more likely to be color-blind. So, also it is not that all animals have the same three cones while humans have 3 cones, night animals have 1 cone, dogs have 2 cones fish and birds have more colour sensitivity and it goes to 4, 5 or in a mantis shrimp goes up to 12 different kinds of cones. So, nature has abundance of how colours are perceived.

(Refer Slide Time: 02:42)

Moving on to how an image is represented, the simplest way to represent an image which you may have already thought of is to represent an image as a matrix. So, here is the picture of the Charminar and if you look at one small portion of the image the clock part you can clearly

see that you can zoom into it and you can probably represent it as a matrix of values in this case line between 0 and 1 and obviously you will have a similar matrix for the rest of the image too.
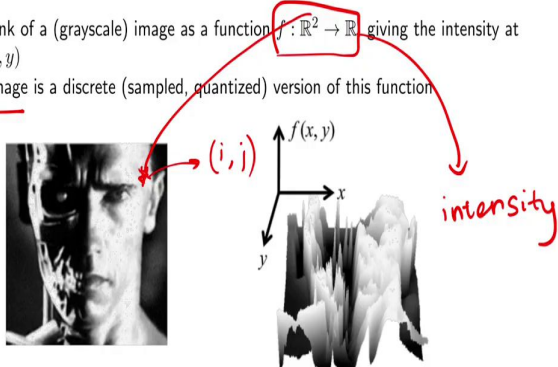
So, but is very very common in practice while we are talking here about using it with values 0 to 1, in practice people use up to a byte for representing each pixel which means every pixel has can take a value between 0 and 255 to byte and a in practice we also normalize these values between 0 and 1 and that is the reason why you see these kinds of values in a representation.

And also, to keep in mind is that for every colour channel you would have one such matrix if you had a Red, Green, Blue image, you would have one matrix for each of these channels. What would be the size of this matrix, the size of these matrix would depend on the resolution of the image. So, recall again what we spoke about the image sensing component in the last lecture, so depending on what resolution the image sensor captures the image in, that would decide the resolution and hence the size of the matrix.

(Refer Slide Time: 04:27)

A matrix is not the only way to represent an image, an image can also be represented as a function, why so? It just helps us have operations on images more effectively if we represent it also as a function, certain operations at least. So, in this case we could talk about this function being going from R^2 to R, where R^2 simply corresponds to one particular coordinate location on the image, say (i, j) and that is what we mean by R^2 .

And the range R is the intensity of the image that could assume a value between 0 to 255 or 0 to 1 if you choose to normalize the image. And a digital image is a discrete sampled quantized version of that continuous function that we just spoke about, why is it a sample quantized version by sample we mean that we sample it at that resolution, originally the function can be continuous which is like the real world in which the image was captured.

Then we sample the real world at some particular pixel values on some grid with respect to a point of reference and that is what we call as a sample discrete version of the original of the original continuous function. Why quantized because we are saying that the intensity can be represented only as values between 0 and 255 and also in the same steps you cannot have a value 0.5 for instance at least in this particular example. Obviously you can change it if you like in a particular capture setting but when we talk about using a byte for representing a pixel you can only have 0, 1, 2 so on and so forth till 255 you can not have a 0.5 so you actually discretized or you have quantized the intensity value that you have in the image.

(Refer Slide Time: 06:25)

$$\hat{I}(x,y) = I(x,y) + 20$$
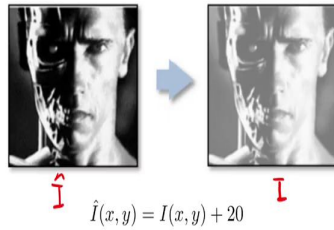
So, let us talk about transforming images when we look at them as functions, so here is an example transformation so you have a face and we seem to have lightened the face in some way. What do you think is the transformation here? Can you guess? In case you have not, the transformation here is if your input image was I and your output image was $\hat{I}$ you can say that $\hat{I}$ hat is $I(x,y)$ plus 20. And 20 is just a number if you want it to be more lighter you would say plus 30 or plus 40, again here we assuming that the values lie between 0 and 255.

(Refer Slide Time: 07:11)

Image Transformations

$$\hat{I}(x,y) = I(x,y) + 20$$

One more example, let say this is the next example where on the left you have a source image on the right you have a target image. What do you think is the transformation, the transformation is $\hat{I}(x,y)$ would be $I(-x,y)$. The image is reflected around the vertical axis, y axis is fixed and then you rotate, you flip the x axis values. If you notice here both of these examples, the transformations happen point wise or pixel wise, in both these cases we have defined the transformation at a pixel level. Is that the only way you can perform a transformation? Not necessarily.

(Refer Slide Time: 08:06)

Very broadly speaking we have three different kinds of operations that you can perform on an image you have point operations, point operations are what we have just spoken about where
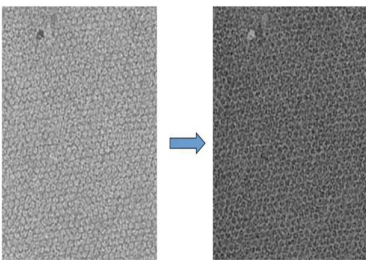
a pixel at the output depends only on that particular pixel the same coordinate location in the input that would be a point operation.

A local operation is where a pixel at the output depends on an entire region or neighbourhood around that coordinate in the input image, and a global operation is one in which the value that a pixel assumes in the output image depends on the entire input, on the entire input image. In terms of complexity for a point operation the complexity per pixel would just be a constant, for a local operation the complexity per pixel would be p square assuming a pxp neighbourhood, local neighbourhood around the coordinate that you considering for that operation. And in case of global operations obviously the complexity per pixel will be N^2 where the image is N x N .

(Refer Slide Time: 09:29)

Let see a couple of more point operations and then we see local and global, so here is a very popular point operation that you may have used in your smartphone camera or adobe photoshop or any other task image editing task that you took on. It is an image enhancement task and we want to reverse the contrast, reversing the contrast we want the black to become white and the dark grey to become light grey so on and so forth.

What do you think? How would you implement this operation? In case you have not worked it out yet, the operation would be at a particular pixel is a point operation so at particular

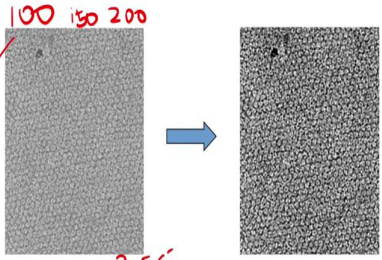pixel (m_0, n_0) your output will be I_{MAX} minus the original pixel at that location plus I_{MIN}, you are flipping so if you had a value say 240 which is close to white generally white is to 255 and 0 is black if you had a value 240, now that is going to become 15 because I_{MAX} in our case is 255 and I_{MIN} is 0. I_{MIN} is 0, I_{MIN} obviously does not matter but this formula is assuming at more general setting where I_{MIN} could be some other values that you have in practice.

(Refer Slide Time: 10:57)

Moving on let us take one more example of image enhancement again, but this time you are going to talk about stretching the contrast when we stretch the contrast, you are taking the set of values and you are stretching it to use the entire set of values that each pixel can occupy, so you can see here this is again a very common operation that you have used if you edited images.

What do you think is the operation here this is slightly more complicated then the previous one, in case you already do not have the answer, remember let us first find out the ratio so you have a typical $I_{MAX} - I_{MIN}$ which is 255-0 by max of I in this image minus min of I in this image let us assume hypothetically that this image on the left had its max value to be 200 and its min value to be 100.

If that is the case this entire ratio here that you see is going to become 2.55 this is (255-0)/100. So, you are simply saying that I am going to take the original pixel whatever let us assume for the moment that the original pixel had a value, say 150 so if this had the value

150 so you subtracting a minimum, so which means you have a value the minimum is 100, so you are going to have 50 into 2.55 plus I_{MIN} for us which is 0 which roughly comes to 128. So that is 50 percent of the overall output.

So, what was 150 which was in the middle of the spectrum in the range of values that we had for this input image now becomes 128 which becomes the middle of the spectrum for the entire set of values between 0 to 255, you are simply trying to stretch the contrast that you have to use all the values that you have between 0 and 255. Which means what would have gone from dark grey to light grey now goes from black to white that is how you increase a contrast, so this is called linear contrast stretching a simple operation again but in practice we do something more complicated.

(Refer Slide Time: 13:27)

So, we do what is known as histogram equalization you may have again heard about it perhaps used it in certain settings if you have heard about it, read about it and that is going to be your homework for this particular lecture.

(Refer Slide Time: 13:41)

So, let ask the question do point operations satisfy all the requirements we have of operating on images? Let us take one particular example, so we know that a single points intensities influence by multiple factors we talked about at the last time and it may not tell us everything so because it influence by light source strength, direction, surface geometry, sensor capture, image representation and so on.

So, it may not be fully informative so let us take an example to show this, so let us assume we give you a camera and you have a still scene no movement how do you reduce noise using point operations. The noise could be cause by some dust blowing in the scene could be cause by speck of dust on the lens of your camera or by any other reason for that matter could that they could be a damage on one of the sensors.

Noise could be at several, at various levels, how would you reduce noise using only point operations? The answer you have to take many images and average them because it is still scene, we can keep taking images and hope that the noise gets averaged out, across all of your images that you took and you take the average of all of your images it is a bunch of matrices you can simply taken element wise average of all of those matrices and that can help you mitigate the issue of noise to some extent.

But clearly that is the stretch you do not get multiple images for every scene all the time and you do not get a still scene that is absolutely still all the time to there is always some motion

and so this may not be a method that works very well in practice. So, to do this we have to graduate from point operations to local operations.

(Refer Slide Time: 15:31)

So, let see what a local operation means, as we already said a pixel value at output depends on an entire neighbourhood of pixels in the input around that coordinate whichever coordinate we want to evaluate the output at.

(Refer Slide Time: 15:47)

So, here is a very simple example to understand what local operation is standard example is what is known as the moving average, so here you have the original input image I as you can see the input image I is simply a white box placed on a dark grey background or in this case a black background because you can see zeros as the values assume that that means a black background.

So, the image has the particular resolution in this particular case it is a 10×10 image and the white box is located in a particular region. But the problem for us is we are going to assume that this black pixel in the middle here and this white pixel here are noise pixels that came in inadvertently. So, how do you remove them?

So, the way we going to remove them is to consider a moving average, so you take a 3×3 window, need not be 3×3 all the time could be a different size, further moment we are going to take it 3×3 and simply take the average of the pixels in that particular region. So, the average here comes out to be 0, so you fill it at the center location of that box.
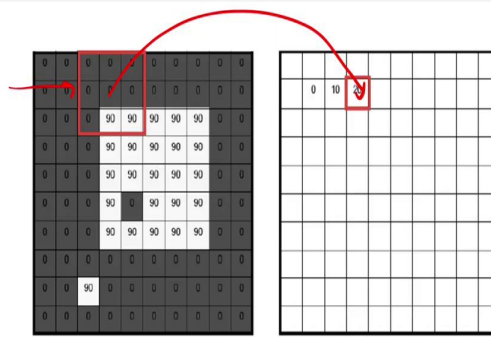
(Refer Slide Time: 17:03)
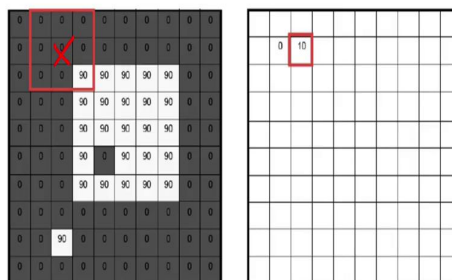
Credit: Steve Seitz, Univ of Washington

Local Operation Examples: Moving Average

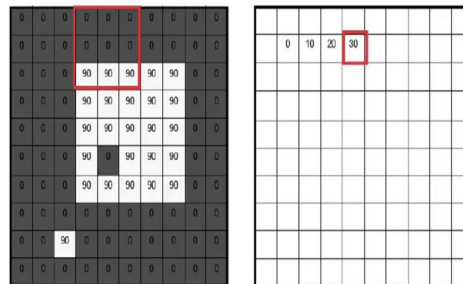Credit: Steve Seitz, Univ of Washington

Vineeth N B (IIT-H) §1.4 Image Representation

Moving on you now move the 3×3 box to the next location you again take an average now the sum turns out be 90, 90/9 = 10. Similarly, move the box slide the box till further and once again take the average of all pixels in the box in the input and that gives you one value in the output. Clearly you can see that this is a local operation, the output pixel depends on a local neighbourhood around the same coordinate location in the input image.

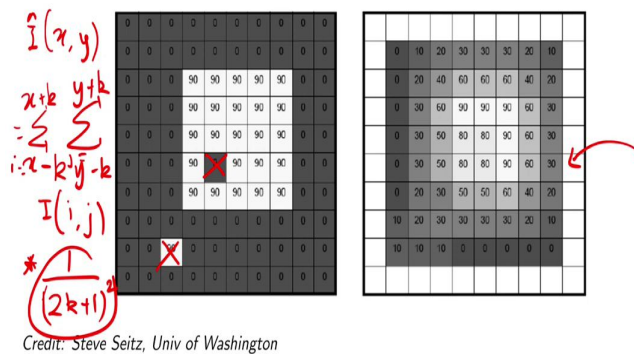(Refer Slide Time: 17:46)

Local Operation Examples: Moving Average

Credit: Steve Seitz, Univ of Washington

Vineeth N B (IIT-H) §1.4 Image Representation

Local Operation Examples: Moving Average

Credit: Steve Seitz, Univ of Washington

And you can continue this process and you finally will end up creating the entire image looking somewhat like this, so you can see now you may have to squint your eyes to see this, you can see now that the seemingly noise pixels here and here in the input have been smoothened out because of the values of the neighbours and the output looks much smoother, here is a low resolution image so it looks a bit blocky.

But if you have higher resolution it would look much smoother to your eyes. So, what is the operation that we did, let us try to write out what operation we did. So, we said here that $\hat{I}$ at a particular location say (x, y) is going to be, you are going to take a neighbourhood. So, which means you going to take the same location in your input image and say you are going to go from say x minus some window k to x plus some window k.

Similarly, we are going to go from some y-k to y+k and let us call this say i, let us call this say j you are going to take the values of all those pixels in the input image. And obviously we are going to average all of them, so you are going to multiply this entire value by 1 by because the neighbourhood goes from x-k to x+k there are totally 2 k+1 pixels there.

So, you are going to have $(2k + 1)^2$, because for x will have 2k+1 pixels, for y you will have 2k+1 pixels and then you just, so the total number of pixels is going to be 1 cross the other and in this particular example that we saw k was 1 for us, we went from x-1 to x+1, so if you took a particular location on the output, you took the corresponding location on the input and

then one to the left and one to the right. So, from x-1 to x+1, y-1 to y+1 and that creates a 3×3 matrix for you and that is what we are going to finally normalize it by. That becomes the operation that you have for your moving average, so this is an example of a local operation.

(Refer Slide Time: 20:11)

Moving to the last kind of an operation called a global operation as we already mentioned in this case the value at the output pixel depends on the entire input image. Can you think of examples?

(Refer Slide Time: 20:27)

Credit: Mathworks MATLAB Toolbox

In case you have already not figured out, a strong example of something like this, this what is known as a Fourier transform we will see this in a slightly later lecture but there are other operations to that can be global depending on different applications, we will see more of this a bit later and we will specifically talk about Fourier transform a bit later.

(Refer Slide Time: 20:49)

Homework

Readings
- Chapter 3.1, Szeliski, *Computer Vision: Algorithms and Applications*

Questions to Answer
- What is histogram equalization, and how do you derive its formula?

Vineeth N B (IIT-H) §1.4 Image Representation

That is about this lecture so your readings are going to be chapter 3.1 of Szeliski's book and also as we mentioned think about the question and read about histogram equalization and try to find out how it works and what is the expression you would write out to make it work.