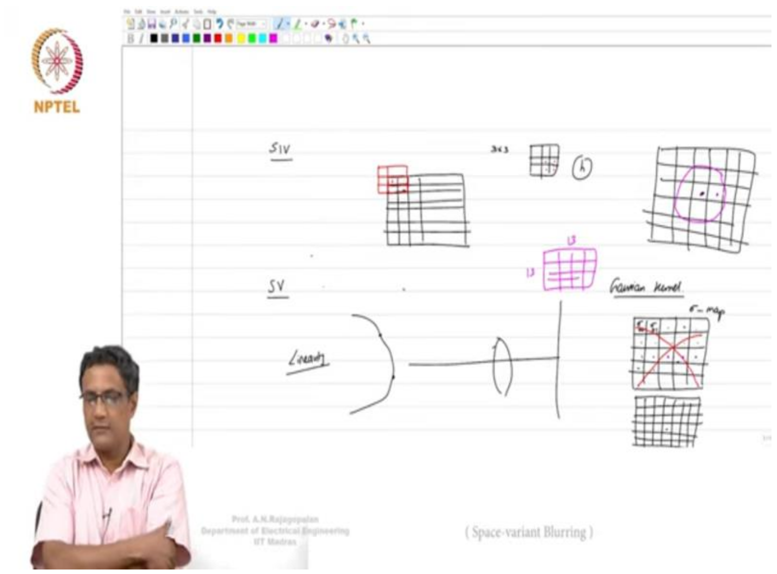
Image Signal Processing
Professor. A.N. Rajagopalan
Department of Electrical Engineering
Indian Institute of Technology, Madras
Lecture No. 25
Space-variant Blurring

(Refer Slide Time: 00:16)



So, we are yesterday I was talking about this space invariant space invariant invariant case which is which is of course simple. It is like your normal kind of a convolution except that at the boundary so I will just say this tell you what I meant. See you have something like this. This is your image and I was telling that saying that here is your image and if you keep the kernel here at the boundary.

So as long as, let us say it is a it is a 3 cross 3, this is your H which could have come from a gaussian or a pill box or whatever and if this guy sits inside, then of course, you do not have an issue. Here there is no issue because it is fully this kernel but when you go here, it does not sit fully because then it will come out like that, look like that.

So, now what happens is you have only 4 intensities is to average. If you if you simply take so if these are those 4 weights, in this kernel and if these are those 4 weights and what it will mean is you simply do you simply add you simply weight each of these intensities below with that weight and add up but then what can happen is because these weights on sum up to 1 because of the entire thing will sum up to 1 but not just the some isolated for weights.

So, what will happen is if let us assume that these values are all equal 40 40 40 40. Then if you just just use this weighted average then it will actually be less than 40 which is which is not which is not because of because of the blurring or anything. It is simply happening because of the fact that you are using underutilizing the weights. So, that is why I said that you could normalize these weights at the boundary. So, take these 4.

Suppose they are ABCD, add them up and then scale each each one of them by, you know, A plus B plus C plus D so that these sum up to 1, the boundary and then you just apply it. Just I mean, so there are different ways to do it, the boundary people do not worry so much about what is happening at the at the at the last row and the last column or last 2 rows and last 2 columns and so on because rest of the image is still so very big looking at 1024 so 2 rows and 2 columns we are not so bothered.

But if you are if you are interested in, what you could do perhaps is either you can go with it as it is, but then that can reduce intensity. So, instead you could just renormalize the kernel only at the boundary but do not do not use this elsewhere. Elsewhere, it will be the actual kernel. Only the boundaries renormalize and use it. The other one is the space variant blurring. This is more this is more tricky and some people call it space variant convolution, but I do not know.

See convolution itself means like it should be invariant time invariant or space invariant but these are terminologies that has come to be accepted. Some people say it is space variant convolution, but we will simply say space variant blurring. So, here what happens is, we would like you to get a gat a feel for how an image would look if a camera if the lens was looking at actually a 3D scene.

So, so you have you have a you have a lens here and then you want understand what will be the what might be the image formation now. How could this image be interpreted? Just like the image that I showed you yesterday where there was a, what was that, there was a book rack when then the blur was kind of increased rate backwards. So, you will not understand probably how this service image formation is possibly happening.

The only thing is this h that you are taking, normally when you synthesize you will assume some h, you will assume some simple model and do it but in reality h could be it could be something else and then this is strictly a Gaussian and so on. So, we do not worry about finding the exact h for a lens because for every lens, h will keep changing. So, we just assume that you can be reasonably model for the same reasons that I said yesterday and you could model it as a gaussian and come to some approximations what might be going on but if somebody gave you the actual h you could use that.

Now in this case, because just to simplify matters what we do is, we will, so here, what we will do is, we will, we actually give you give you a surface say what you need is, first of all, the point is because of the fact that the scene is 3D now so what it means is that every point is that is that is that some Z from the lens.

So, each has its own sort of a depth. So something is nearer, something is farther and so on and something could be at the plane of focus in the sense that some points could actually be coming into focus on the plane. So, in order to in order to capture all of that, so what so what we do is the following? We ask you to assume a gaussian kernel everywhere ask you to assume a gaussian kernel at all the places and we give you a we give you a distribution of the sigma over the image. So, if this is your image assume that it is some 5 cross 5.

Then we will give you so actually what we do is we give you give you a sort of a gaussian surface itself. The kernel is gaussian is one thing the surface also, just for analytical lease we give you a gaussian surface, which actually means that this surface if you try to plot this, it will look somewhat like that. So, you can you can think of some kind of dome that is kind of sitting and the cameras looking at this kind of a dome-like structure. So what so what we do is, we actually give you an analytical function which is simply a gaussian actually which will allow you to find out sigma at all places.

So, it is like saying that saying that, you can find out sigma, let us call this sigma 0 0, we will be able to sign out sigma 0 1 so ideally sigma can vary at every pixel now. In the earlier case was where sigma was a constant all over the image, so there we had the same sigma wherever you go. Therefore we will use the same kernel we said a convolution is fine but now what will happen is because of the fact that every point is at a different depth or can be at a different depth so and in this case they are.

So, we actually give you a distribution which you can calculate with some with some initial conditions with some boundary conditions we give you so that you can actually find out what is the sigma at all the places. Can you make it when I write this? You can. So, so so so you find out at all the places and now now that you know the sigma map so this is like your sigma map and it will have a resolution that will be identical to that of the actual image that you want to blur now. Now, shift invariance will not hold because of the fact that things are addressed in different depths.

So the only thing that you can exploit now is linearity, because the lens is still still linear. So like I told you, linearity is independent of we all know that it is of any space invariance and so on. So, if I have one point light source which sends out a certain sort of a blur circle then I could have I could have another point sending out its own its own blue circle because it is at a different depth and then superficial holds because the lens is linear.

Therefore, all that we need to do is add them all up right on the same image plane. So that as you can guess what would you do. So, you would actually you would take up, you will take a grid like this. This is your output G. In the earlier case, what we would have done is you would adjust computed these convolutions locally and then simply you would have entered those values into this this array which gives you an output G. Now, you cannot cannot do that anymore, you cannot do a local weighted average and all because the blur colour itself is continuously changing.

So, what would you do? You would actually find out that for this suppose it said that I want to find out what I need to do here, somewhere here. Let me just put that in some colour. So, I want to find out in the in the output array what should I enter there? So, what I would enter there is I find out from my sigma map whatever is that value of sigma here because because I have the sigma distribution. So, I will have a value for sigma there, let us say that it is 2 or something. This is an example.

Then what I will do? I will find out a niche for that 2. So, it will be like 6 sigma plus 1 whatever so you will get a sort of 13 plus 13 kernel. So, we will go from minus 6 to 6. So, although although h mn, you will actually compute so that you get the whole kernel. Then what would you do? Just as you go back to that the equation when we wrote g mn is double sum f n prime m prime h mn semicolon n prime m prime where it depends upon where your delta is getting applied. So, here this equality mean that kernel you should multiply with the with the intensity in the actual original image at that point.

So, if let us say this is the sigma map and let us say you also have the image which is of the same size as this correct, imagine that you're your image is this. This is your focused image that you want to blur and of course we will also give you the focused image because you need that to kind of recreate this blurring effect. So, at the same location you go here, at this location, you have some f of whatever and at this location we are let us say you are some impression of 120.

Then 120 will multiply this kernel. So, you have this kernel that now you have for that sigma. This is a 13 plus 13 kernel whose weights will sum up to 1 all of that it will follow. Now, you multiply this sigma not sigma, this intensity by this and then whatever and so here you will get a 13 cross 13 grid on the in the output array a 13 cross 13 sub-array, you will get multiplied by 120 so that will sit here. So, that will sit somewhere here as a 13 cross 13.

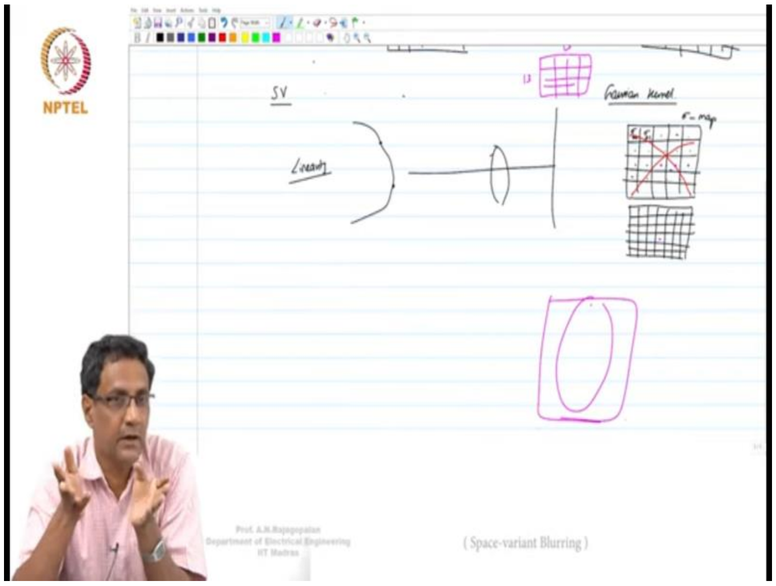
Then you move to the next pixel let us say here. What should we do here? Now again, I will compute what is the sigma here I will compute the kernel corresponding to that. I will multiply that kernel by the intensity here and then and then simply simply write and simply simply what we call, you place that you place that at this particular location. So, what it will mean is you will keep on adding this array and it is like saying the array did not have any values at all. All 0.

Then then you put 1 kernel, then you add with whatever was your earlier array, so you will have only let that kernel multiply that intensity then then you then you then you put another kernel on top of that, add it with the with the earlier result. Keep on doing this all over the image. So, this is simply this linearity property. You cannot use shift invariance because that would be then wrong.

So, this is how you will have to generate now the entire enter sort of a blurred image. Do you have any doubt on this because this sometimes is a little confusing? So, I thought explain to you what how it works. You see that you cannot you cannot take this, you cannot do a local averaging like we do in a sort of a convolutions. All that we can do is treat each point light source is independent and then see what it will shine on the on the image plane. What will it what will it result in and what it say what it will give is simply the original focused image intensity at the point multiplied by that kernel.

That is how we did no? The blur circle and all. That is how we came up with the with the blue circle thing. Instead of a circle now, it will be a kernel of a certain size and then has a distribution. Here, we are assuming it to be a gaussian and we are also telling you what that sigma is. In in real, in in in a kind of a real situation, this will all happen automatically. So, the lens will have its own characteristics and then it will automatically do all of this and then what you see is finally a resultant image that you see is blurred space variantly and all that. Just that you can actually get a feel for it if you do it yourself.

(Refer Slide Time: 12:23)



Now the point is that at this point we could switch to some other topic but but but then the point is, what I feel is having walked so much in terms of how and how the lens and how the image is formed using a lens and so on having understood all of that, now is the right time to actually extend this idea a little further. One is to understand how this image formation happens and now what basically one could do is, one could now sort of play the other role in the sense that if see a blurred image something like that.

So so here, I have my image plane and then I see a kind of a blurred image there. Is there is there something in that image that I can extract in order to be able to tell something about the scene now. So right now, what we did was more like a forward problem. A forward problem is where you assume that that right a certain 3D scene exist. You assume that that there is a certain point spread function, that is whatever whatever form it is changing and then you say that for for this imaging system what will be the output. Correct?

Now a more interesting problem, not that the other one is not interesting, producing a more interesting problem is, if I showed you the blurred image, is there a cue out there that you can use. Cue is like clue. Is there a clue out there in that image that will actually tell you something about the scene itself? So, that is called an inverse problem and inverse problems are always harder because the forward problem is typically very very well posed whereas inverse problems are typically not well posed.

So, a lot of effort has gone into this and a lot of effort has gone into so this whole area by the way is called comes under. So, this is really a vision kind of a problem, a vision problem but since you are so close to this, I just thought, I feel it is interesting to just see how this can be applied along the way. So, this is called shape. This whole area is called shape from x.