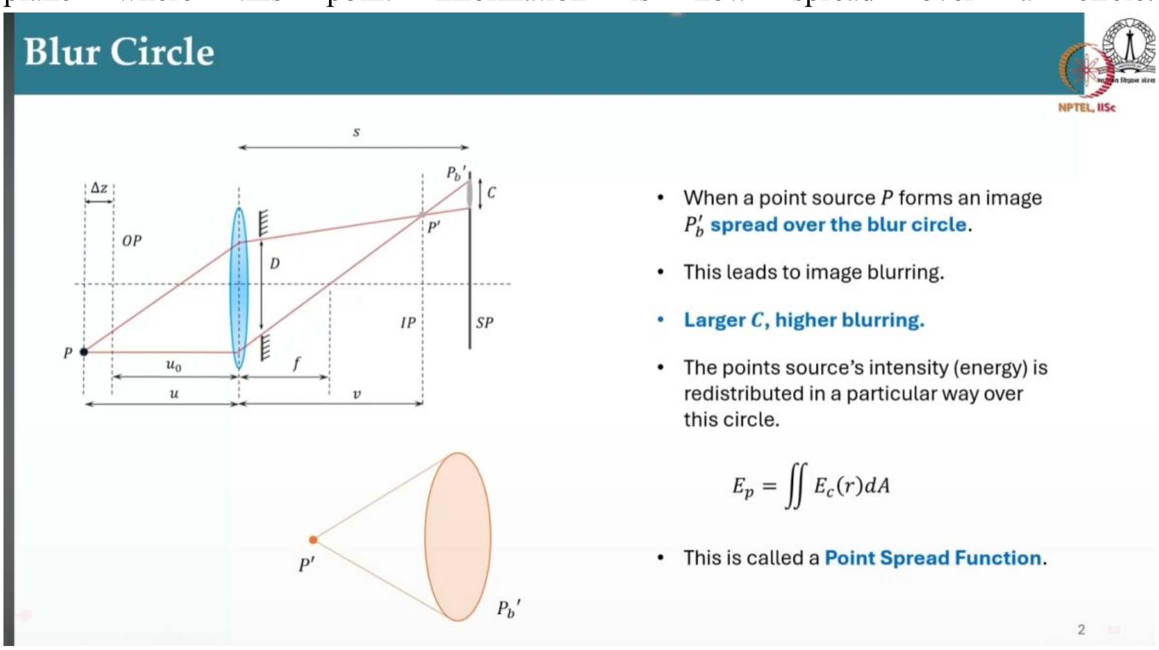
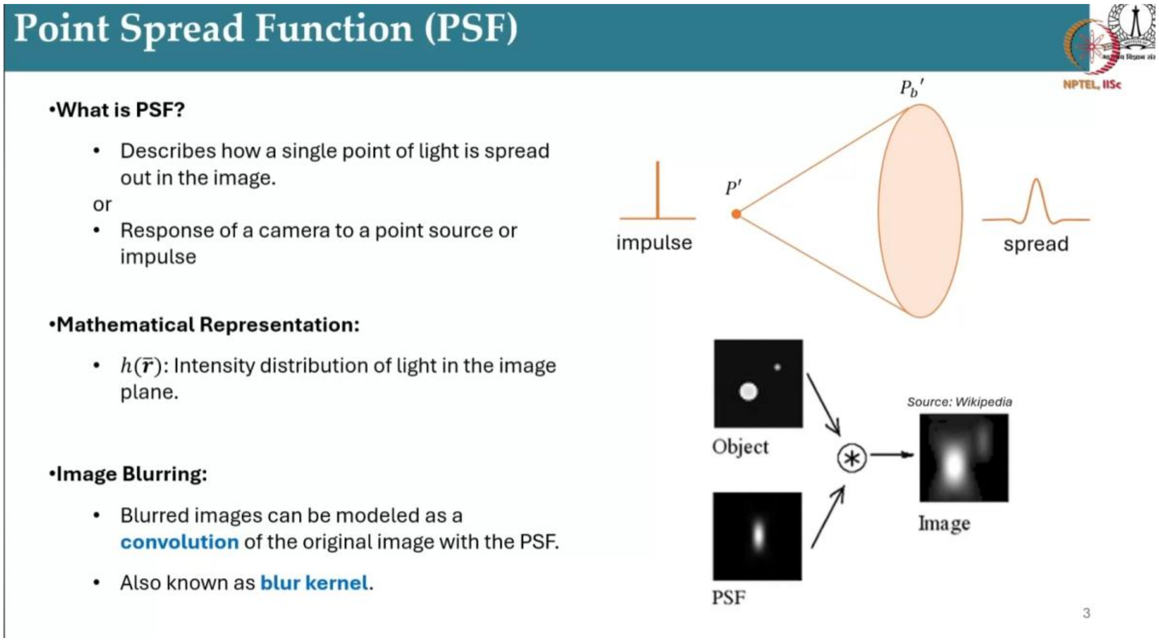
Advanced Measurement Techniques in Fluid Mechanics and Heat Transfer
Prof. Saptarshi Basu
Department of Mechanical Engineering
Indian Institute of Science, Bengaluru
Week – 06
Lecture - 28
Depth from Defocus – 2

Hello, everyone, and welcome back. In the last lecture, we discussed the fundamentals of blurring from a ray tracing point of view or the geometric optics point of view. We have found something about a blurred circle, and we have seen that a point source is spread over that blurred circle, which is representative of blurring in any imaging system, and we ended that lecture with something known as a point spread function. So, in this lecture, we will be discussing that in much detail. If we revisit blur circles, we see that if we offset a point away from the object plane by Δ z, the imaging plane is also shifted away from the sensor plane. This leads to the formation of a blur circle over the sensor plane where this point information is now spread over a circle.



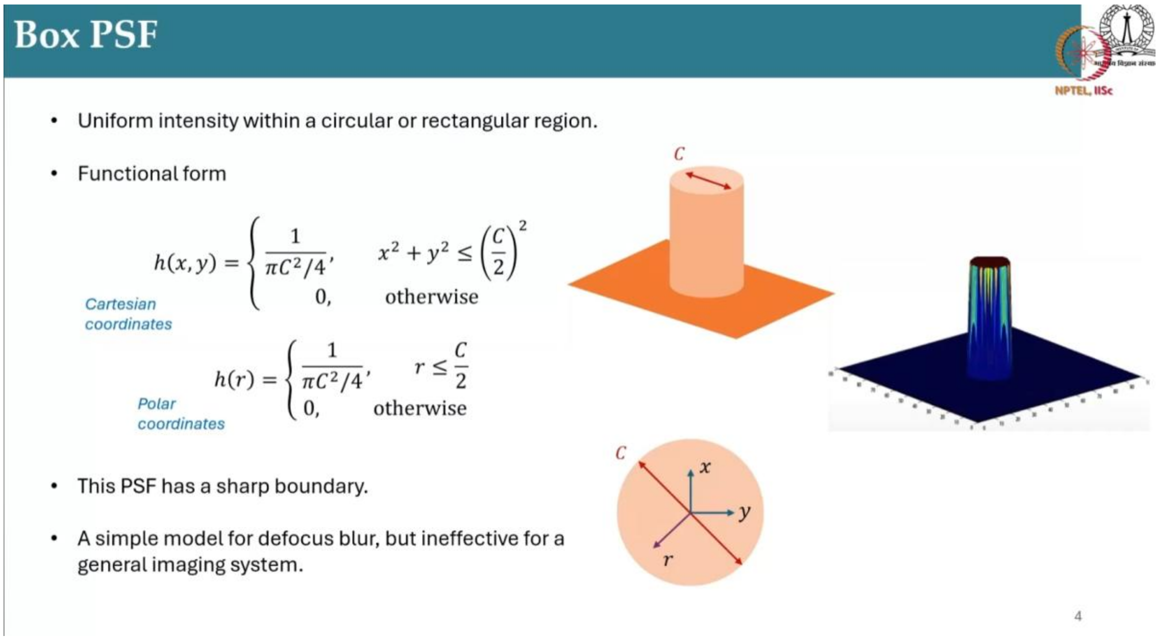
So this point of information forms an image here, but this is now spread over a circle when captured on the sensor plane. If represented in this form, this is just the spreading of information from its like a map from a point to an area. So, we need a function that maps this, and this is something that is known as a point spread function. So, this is nothing, as the name suggests.

This suggests that the spatial distribution of intensity within this circle corresponds to point information. So, if we define the point spread function in a more systematic way, it is something that describes how a single point of light is spread out in an image or how a camera responds to a point source; in an impulse, it looks like a Dirac function kind of thing, and when it passes through our optical system, it leads to a signal that is spread over an area. So, this point spread is defined by the point spread function. Mathematically, it can be represented as a spatial distribution of intensity in the image plane. So, that is something in a way we can define the point spread function, and it is a very powerful tool if we want to describe image blurring.



So, image blurring can be modeled as the convolution of the original image with the point spread function. What do we mean by convolution, which we will discuss soon, and this convolution is actually a very generic term? It is a mathematical operation that can be used in a variety of ways, and sometimes we call this function, which is used to convolute other functions, a kernel. So that's how this point spread function is sometimes known as a blur kernel, and here is a schematic representing that. So this is the object, actually, and this is the point spread function, and we can convolve this object with this point spread function, and you can see this leads to the formation of a blurred image. So here, the image was roughly in focus, and if you artificially convolute it using this point spread function, it leads to an image that looks very similar to a blurred image.

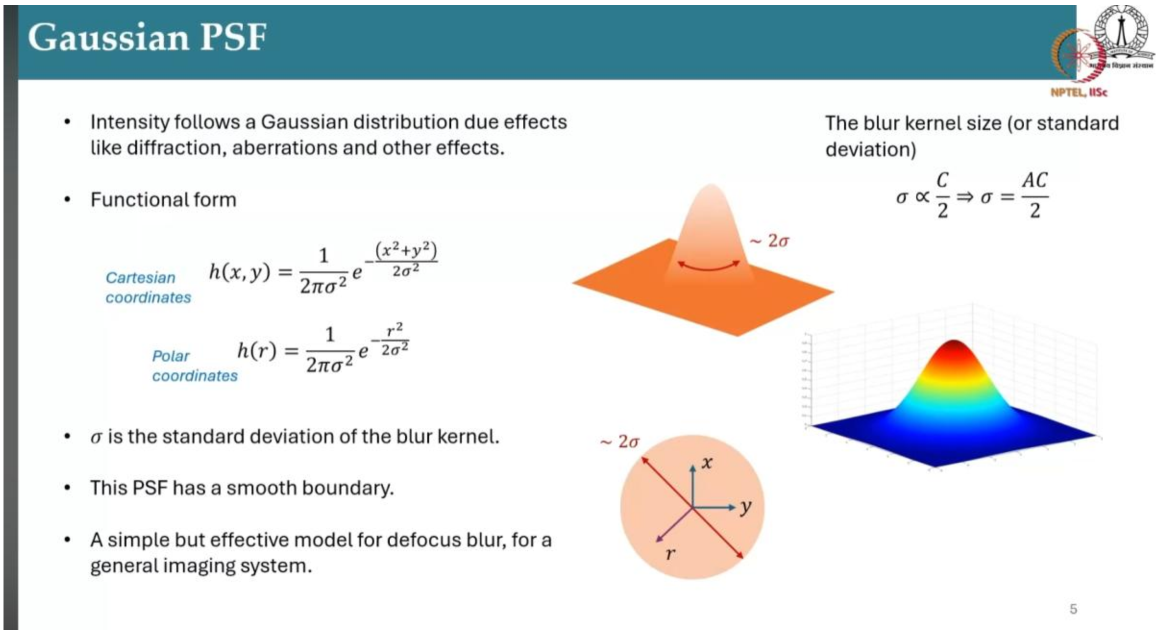
So let us now discuss a few standard point spread functions that are very popular in the literature or in practical implementations. So the simplest one is the box-point spread function. Here it is assumed that the intensity is uniformly distributed within a circular region, and this circular region corresponds to the blur circle. This is represented schematically here, where you can see that the intensity is uniformly distributed within. In a functional form, it looks like this.



So, you have this in Cartesian coordinates. This function H represents your PSF, and you can see here that the value 1 is just normalized by the area of the circle. That's how, if you integrate this over the area, you will get 1. And if you are outside this circle, it's zero. So, this represents that you are inside the circle, and otherwise it's 0.

In polar coordinates, it looks much simpler. If your radius of the radial position is within the circle, you will have this $1/(\pi c^2/4)$, which is nothing but 1 divided by the area of the blurred circle. The problem with this PSF is that, although it is a very simple model that can be used to model this defocus blur, it is ineffective when treated as a general blurring model for any imaging system. Also, this PSF has a sharp boundary, so there is a discontinuity at the end of this PSF, which we do not typically see in any of the imaging systems. But it is still very useful, and it is sometimes used in superposition with some other blurring models to emulate a scenario.

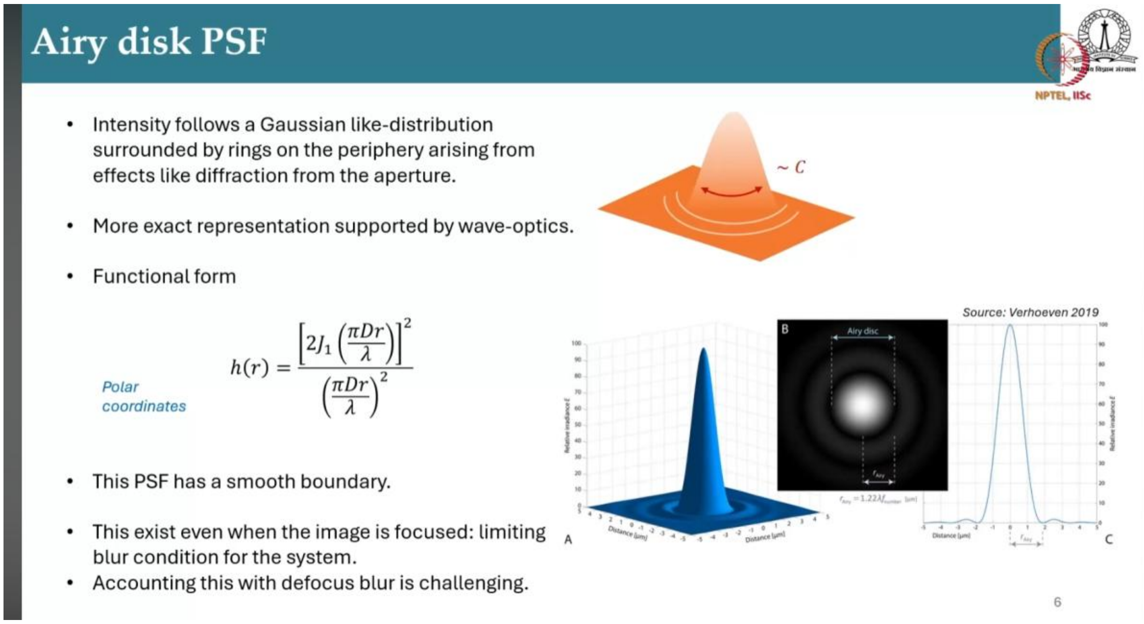
So the next one is the Gaussian point spread function. As the name suggests, in this, we assume a Gaussian distribution of the intensity. As you know, Gaussian distributions are not bound. Their tails asymptotically approach zero. So, basically, it means that they approach zero at infinity.



So, you have to define a parameter that tells you about the extent of the spread, and the standard deviation of the blur kernel or PSF is one such parameter. which appears in this equation. You can see if, in this equation, this r squared appears with sigma squared. So, basically, sigma is something that bounds the kernel in the radial direction, and we treat it as a representative length scale for this blur kernel. So, this blur kernel is smooth.

It has been observed that this blur kernel appears frequently in a lot of imaging systems and hence can be used as a generalized model for a particular kind of imaging system if I put it in a more explicit way. These non-sharp boundaries are due to effects like diffraction or aberrations in your imaging system and other factors. So, this earlier model of the box function is an idealized one. Where you assume that the blur circle is uniformly illuminated, here you see that there is no defined boundary for the blur circle, and that creates a problem in a way because we are assuming a model here that doesn't have any boundary, whereas in the previous ray optics ideologies we have seen that there exists a boundary because there is a cone of light and things like that, so here to bridge those two ideas. We scale the problem in a way that this standard deviation is basically proportional to the blur circle radius, and there appears a constant which we treat as an experimental constant for the system.

So, this is a very effective model which we will use a lot in all of our analysis and we will also discuss in which kind of systems this PSF exists because it is still an approximation. Coming to the area of disc PSF now, it still has a Gaussian-like distribution in the center, but then you will see rings around that center or waves. So, if you look at this schematic, you can see that there is a dome that looks like a Gaussian dome. But then you have ripples around it or rings around it. Those are actually artifacts of diffraction.



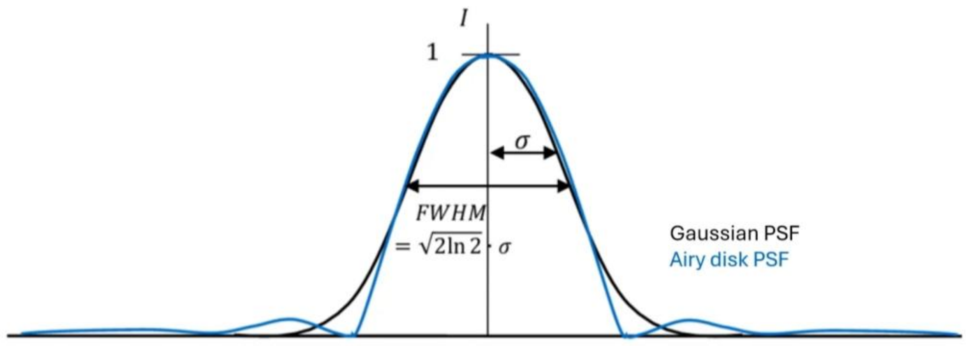
So this is something that is a more exact PSF based on wave optics. Gaussian is just an approximation for the central dome of this, and we assume that the ripples around it or the rings around are weak. So that's why Gaussian PSF, although it is very efficient, is still an approximation. Here you can see that the functional form of the ARID disk PSF looks very complicated. That's why using this directly in systems can sometimes be very tricky.

But, yeah, this will be more accurate for sure. This PSF also has a very sharp and very smooth boundary. And it asymptotically approaches zero at the end. But you will also see some nodes where this value is zero because it's an interference pattern. So you will see a fringe pattern where you will see nodes and antinodes.

The speciality of this is that this will exist even where your system is in focus. So this is a limiting condition for any system. So this is the amount of blood that will be there always in your system. But if you also introduce defocus, then you have to account for both the blurring induced by defocus and also because of this interference that is ARIDISC. So then you have to have some superimposition models for that.

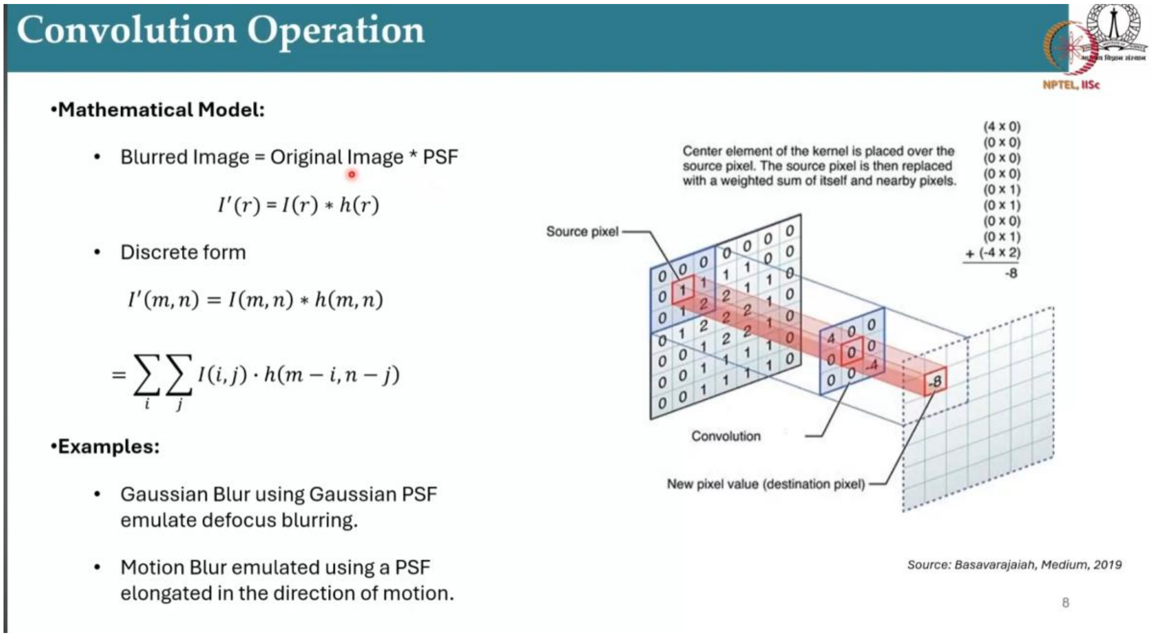
And sometimes modeling this wave optics-based or diffraction-based PSF for such defocused systems is very challenging. So, typically we will stick to a Gaussian PSF in all our analyses for the simpler implementation and design of systems. But yeah, if you are working with coherent light sources, such as lasers and things like that, you cannot avoid these Airy disks and fringe patterns or interferences. Just to compare, here we are comparing a Gaussian PSF with an Airy Disc PSF, and you can see the central dome can be approximated pretty well using a Gaussian PSF. So, that is actually the idea of a Gaussian PSF.

Comparison



So, it is not something that appears from the fundamental equations of optics, but it is an approximation for the central dome of your Airy Disc PSF. which is something you can estimate using wave optics or the interference pattern generated by considering the light rays originating at the edge of the light cone from the aperture, something like that. So, you can refer to any of the optics books, and you can clearly find how this ARRI disk is formed. But yeah, we will stick with the Gaussian PSF. Now we have a model or a functional form of the spread of intensity within a blurred circle.

How do you use that information? We can use that information to emulate blurring. If you have an original image and you want to artificially blur it, you can impose these point spread functions to generate blurred images, and the convolution operation is something that is extremely useful for doing that. So in a functional form, it would look something like this: you have an original image, you convolve it using a point spread function, and you will generate a blurred image. In the discrete form, it will look like this. Why in discrete form? I will discuss this, but you can also represent it in a continuous form, where the summation signs will be replaced by integrals.



So this schematic will be very helpful to explain the convolution. Say you have an array of numbers; this is nothing but a kind of matrix, and you have this convolution operator here, which is also a matrix. So, what you do is treat this convolution as a moving window. So, you place this window aligned with this cell highlighted here and perform the operation such that each of these elements is multiplied by the element of the source. So, the elements of this source image are multiplied by the weights.

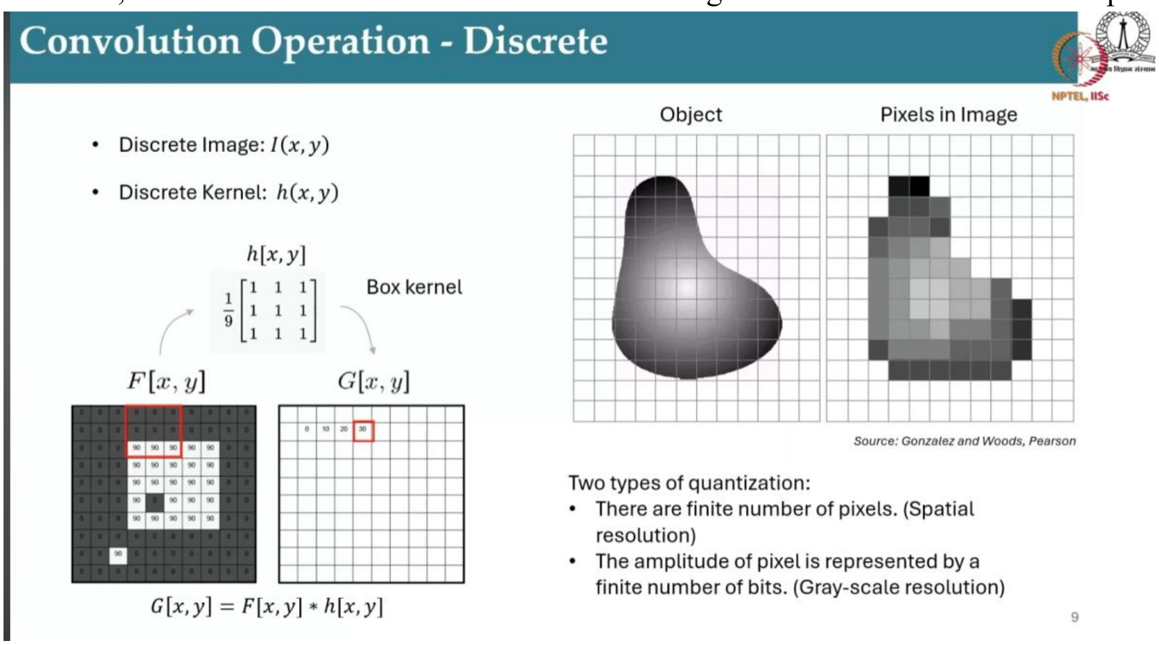
So, you can treat these values as weights and then take the sum. So, it is kind of a weighted summation or a weighted average kind of scenario. So, what you will do is that you will do that. So, this is just an element-wise operation that is represented here for these nine elements that are available. and you will assign the final value to this updated item, like an updated image or updated matrix.

So yeah, it can be slightly difficult to move your brain around this idea, but just give it some time, and maybe you can try this out on a piece of paper, and you will understand it more easily. So you basically do this again and again. Now shift the window to the next cell; do this again. Again, for the next cell, do it again. So what you just do is whatever is occurring within the window, you apply the weights associated with the convolution operator and just take the sum of everything and place it in the final new matrix.

Do this again with the next cell, next cell, next cell. Do it again and again for all the cells, and you will have the convoluted image. So just imagine if. All these values were zero, and the central value was one; then this operation will generate the same thing again. Okay, and whatever I said is represented in a functional form here, so you can break this function down.

For example, you take a four by four, um, say make a 5x5 matrix and a 3x3 convolution operator. You can write element by element whatever is happening, and you can match. if this discretized form is correct or not. So, I guess this will be a good homework exercise to understand this convolution operator. The same thing can be written in the form of integrals if you want to estimate it in a continuous domain.

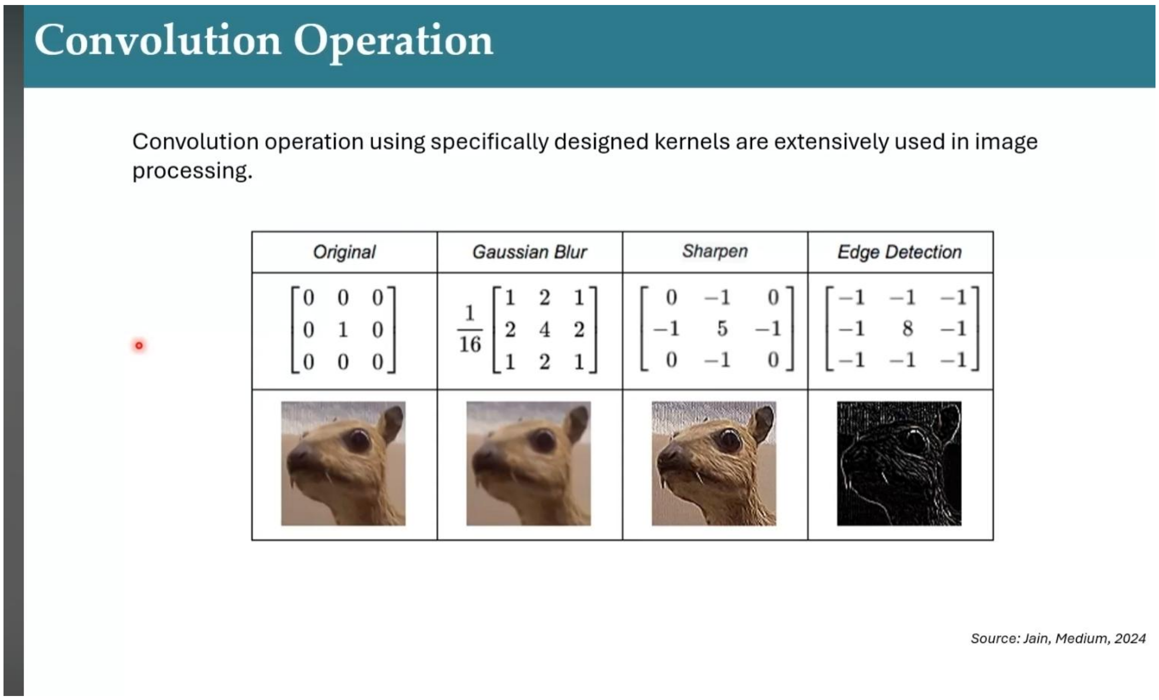
Some of the examples that we are trying to explain here are that you can emulate Gaussian blur using a Gaussian BSF. And this is something in which we are interested because it kind of simulates defocus blurring. You can also generate motion blur that we discussed in the previous lecture using a PSF that is elongated in the direction of motion. So, that PSF is not symmetric. All the PSFs that we have discussed so far were symmetric in the azimuthal direction, but there can be PSFs that are not symmetric in the azimuthal direction, which can lead to a motion-blurring effect in the final output.



Again, coming back to the fact of why discrete: because the images are something that are discrete in nature, it is not a continuous field. This is just the representation of that; say you have an object that is mostly continuous in nature, but when it aligns with the sensor array, which has discrete sensors arranged in a grid, each sensor element will collect the intensity and assign it in a discrete fashion, so your object will look something like this. Most of the time, this captures it in very high resolution, which basically means that the sensor sizes are very small, so the object looks smooth; but actually, it's discrete if you look at the quantized final elements. This is something that is known as spatial resolution. Also, even the intensity that is captured at each sensor level is not.

Continuous. It can be captured at a finite number of bits. This is something known as grayscale resolution. So this is not something that is really important at this stage. So I will not discuss this. But yeah, this is something that is important for you to understand: the information you are capturing through any of the cameras is usually discrete.

That's why we are also talking about this convolution operator in terms of a discrete operator. And here is just a representative of the same thing that we discussed earlier. So you have this image, this moving window marked here, and these weights. So basically, this is kind of an averaging operator, right? You can clearly see that. Here the weights for each of the cells are 1/9.



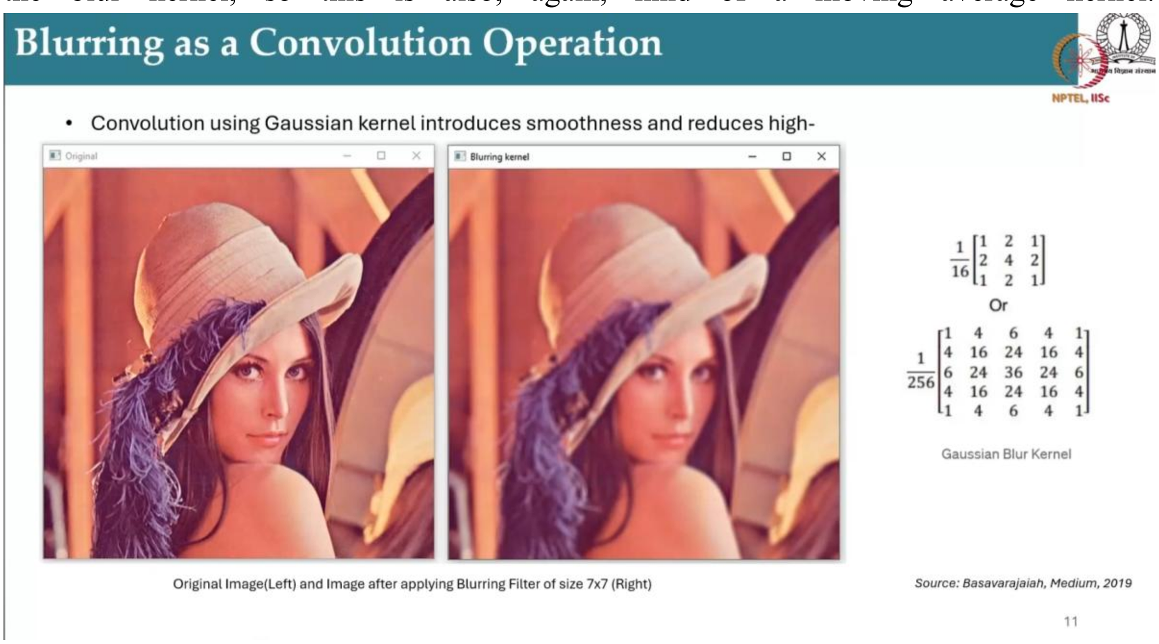
So it is nothing but the sum of all the elements within the window divided by 9, nothing else. So this is kind of an averaging filter. And here, as you can see, all the elements have the same kind of intensity. This H has all the same values, uniformly distributed. So this is a box kernel or box PSF that we discussed earlier, right? And this convolution operator will then be applied to each subsequent cell point by point until the whole space is filled.

And then you will generate a blurred image of the same image. Here are some examples of operators that can be convolved over an image to generate certain characteristics. So, as I have explained, if you take just one at the center and the rest of them as zero, it will just recreate the original image, right? Because you are assigning weight only to that cell, which is again placed back in the new matrix at the same location. So it's like just copying the original image and pasting it into a new matrix. You can have the weights in such a way that the center has the highest values, and as you move away from the center of this matrix or the blur kernel, the value decreases, and it can be done in a very particular way.

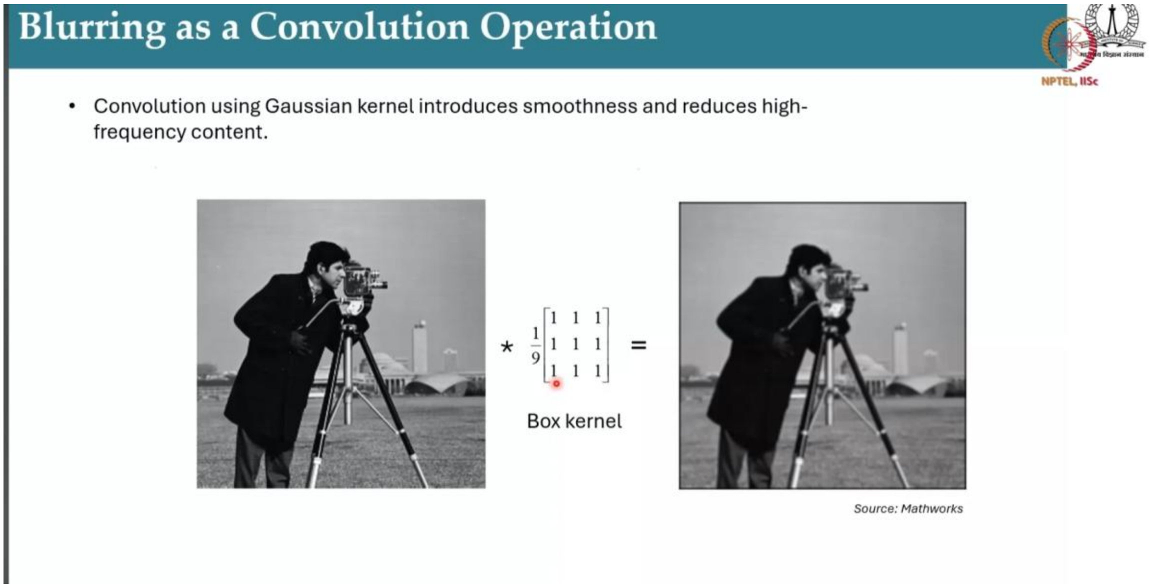
So, in this case, it is done in a way that matches the Gaussian blur kernel that we discussed earlier. So, in the same functional form, you can extrapolate that and get these values. And then you can see that it looks slightly blurred. So, this is the original image.

Here, it looks blurred. This is an operator that sharpens the image. So these are some weights assigned to it, and you can see it leads to the sharpening of the image. You can also detect edges. So these weights, when convolved over the original image, clearly highlight the sharp edges with whiter values, while darker spaces represent the smooth parts of the image. So yeah, this convolution operation using a certain kind of kernels, designed for some particular application, is a very powerful tool in image processing and is extensively used.

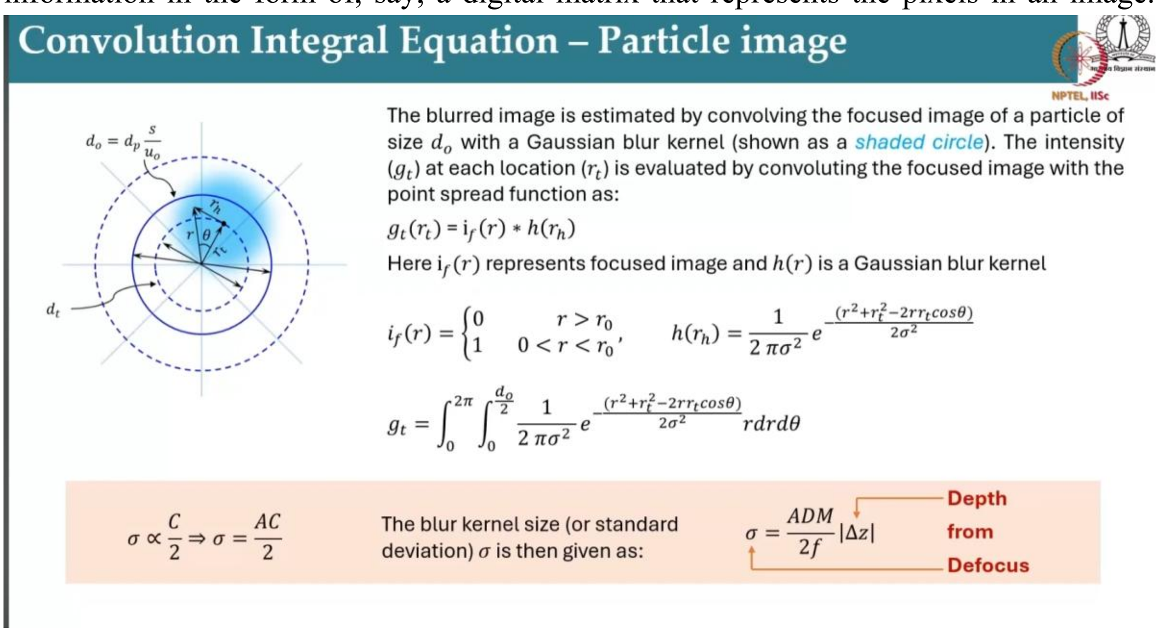
So this is an example of simulating a blur using a convolution operation. Here, a Gaussian blur kernel was used to blur this image, and you can see it looks pretty much blurred. And this smoothing out is controlled by the strength of the Gaussian blur kernel, which is dependent on sigma. The higher the sigma, the greater your extent of blurring. This is an example with the box kernel; here, the intensity is uniformly distributed within the blur kernel, so this is also, again, kind of a moving average kernel.



You can see it leads to blurring in this fashion. You can increase the strength of this kernel by changing the size of the blur kernel; here, the moving average is taken for only nine neighborhood points. But you can take maybe 4 x 4, like 16 neighborhood points, or 5 x 5, that is 25 neighborhood points, for a stronger blur. So here, the extent of blurring is controlled by the size of the blur kernel, as we have discussed earlier. If your blur circle size is larger, your extent of blurring will be larger, so it's the same analogy.



If you take these arrays of one in a 5 x 5 grid... And you normalize it by taking 1/25 so that the net intensity is 1; then it will lead to much larger blurring. Now that we have covered the fundamentals of blurring from the perspective of geometric optics, we have seen the formation of a blur circle. And we have also seen what the role of the point spread function is in defining the redistribution of the point source information or the light information within the blur circle. We will now try to implement it in the specific context of dispersion size measurement. So, we have seen that if we have focused image information in the form of, say, a digital matrix that represents the pixels in an image.



We can apply a convolution operator to artificially emulate the blurring effect. In the same way, we can do that in a continuous fashion as well. From a practical point of view, we do it in a discrete way. Here, in this slide, we are representing that in a continuous way. So, we have this particle of size d_0, and we are seeing how we can, this is the focused particle image, and how we can convolute it using a Gaussian blur kernel.

To generate a blurred particle image, these are just the mathematical forms of all those things, and when we finally substitute everything, we get this integral equation, which is crucial in certain methods or certain forms of the DFD technique. By DFD, I mean depth from defocus, so as we have seen while defining the Gaussian blur kernel, that we can. Approximate the standard deviation of the blur kernel with the blur circle radius, so we have put a proportionality sign here correlating both, and we can then have an experimental constant, capital A, which will be dependent on a lot of factors, some of which might not be deducible from theory. What it does is provide a relation of this blur circle diameter with respect to the Offset of the particle from the object plane or movement of the particle in the depth direction now becomes a foundational equation correlating depth, which is Δ z. We do defocus, that is, sigma, because sigma is representative of the amount of blurring or defocus, and Δ z represents the position.

So in this form, with all the discussions that we have already had. We brought everything down to our motivation depth from defocus, where we are trying to determine the geometric parameters associated with the particle by estimating defocus in an image. So, that is what our motivation is. In the next lecture, I will be discussing in detail this particular convolution integral equation. Here, I am just browsing through it, but there I will depict two steps of how we obtained this equation and how it is useful.

We will define certain elements, such as thresholding, Fourier transforms, and other concepts. The tools that are used along with this idea to implement the depth from defocus to estimate particle sizes. Thank you.