**Optical Methods for Solid and Fluid Mechanics**
**Prof. Aloke Kumar**
**Department of Mechanical Engineering**
**Indian Institute of Science- Bangalore**

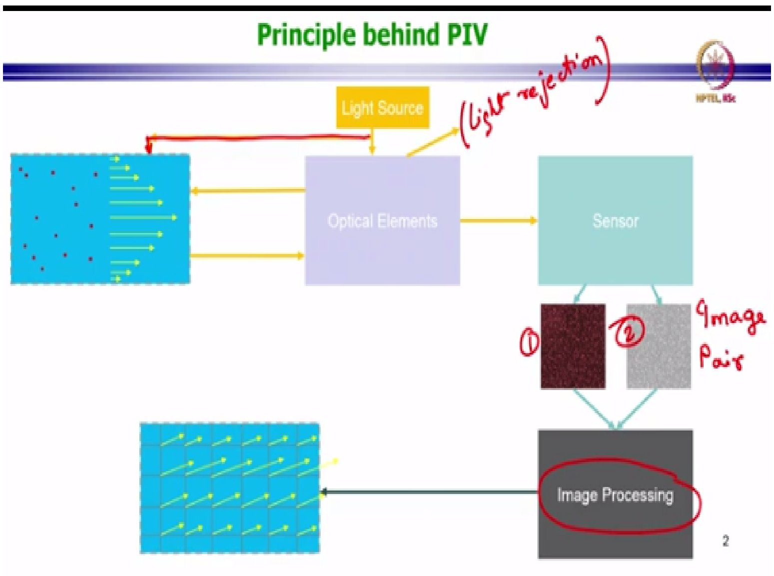
**Lecture - 19**
**Particle Image Velocimetry IV**

Hello and welcome back. We have been seeing the details and the principles behind an experimental operation of particle image velocimetry. We have already seen quite a bit. So we have discussed about tracer particles, the need for them and what are the different types in different cases. We talked about lasers and light, options for particle image velocimetry.

We have also discussed aspects of optical elements, although we have not discussed enough and the reason is that to discuss optical elements, one has to have a strong side off shooters to take off into geometric optics and elements of optics, which is not part of this course. And the particular optical issues vary between different situations.

So we are not going to be able to go into the depths there. But we have familiarized ourselves with at least some aspects of a practical situation.

**(Refer Slide Time: 01:11)**




I want to go back to this particular diagram that I had shown you about the principles behind PIV and I want to introduce a couple of small corrections to this particular diagram. So this is a newer diagram. In the past lectures, you saw a slightly older version of this. Here what I have done is I have just made sure that this arrow was drawn from the optical elements to the light source.

And here I have just drawn it off to the side. So this is basically to signify that this is light that is rejected by the optical elements. So not all light that goes from lights from the light source enters the entire optical train, some of the light can be rejected also. For example, what happens when you put light filters, a light of the light is rejected before even enters the optical train. So this signifies light rejection.

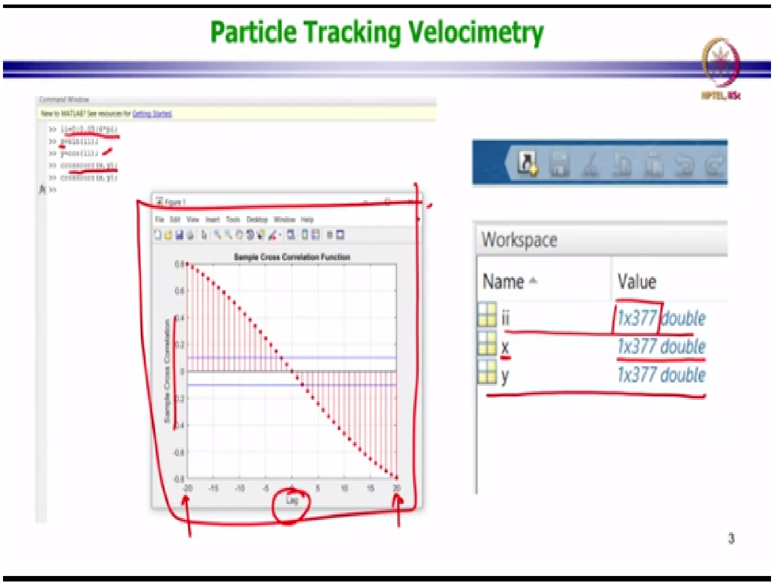
And I have also drawn this particular line here to signify that your light source need not always light or the light need not always go through an optical element train in order to eliminate the tracer. It is possible specifically in natural lighting cases that you have the light source which is directly illuminating the tracer particles and then you have the light going through an optical element into your sensor, right? Okay.

Now that brings us to this particular aspect of particle image velocimetry which is probably the heart of particle image velocimetry which is taking two images and this is also often called the image pair. We will see why it is called an image pair, okay? So this is an image pair 1 and 2, two images which are taken within a very small time interval of each other.

This is why by the way, where you require a double pulse laser often. For fast flows, you need to eliminate your tracer particles very quickly in succession for the first image or the second image. These two images are then sent to a processing unit, which by the way this entire processing can be done at a much later stage. So usually in a lab environment, you will be doing this particular experiment.

You will be collecting all the image pairs that you require. And then you are going to do this image processing much later. Perhaps it obviously depends on the particular situation. But image processing need not be done simultaneously with the rest of the experiment. Once you do the image processing, you will end up with a velocity diagram like this, right? So we are now going to start and take a look at the different aspects of image processing.

**(Refer Slide Time: 03:59)**




Before we do that, there is a mathematical element that I just want to recall and familiarize ourselves with one more time. And I had talked about cross-correlation, right? So in the mathematical preliminaries, when the mathematical preliminaries were being discussed, we discussed the idea of something called cross-correlation between two signals.

So what I have done here, these are screenshots from small code that I was trying to write. And what I have done here is that I have defined a particular variable that goes from 0 to 6 pi from this interval, this is written in MATLAB. But you may use other softwares for this obviously. So you if you have access to some software, which helps you do these calculations, then do please go ahead and try this out.

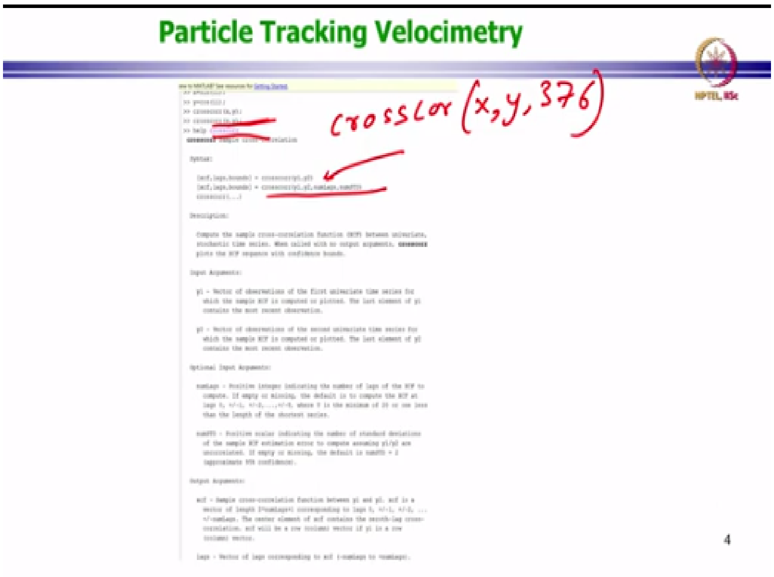
And so I have defined a particular variable which now basically is a matrix. So my ii is a matrix of double values. It has 377 entries. So it is a basically a vector. Using this vector, I define another vector x, which is sine of ii. So you can see my x, which is the sine of ii. This is the workspace, by the way of MATLAB. So I have taken a screenshot of that, and I have magnified it.

And you can see this also has 377 entries. And I have defined another variable called y. And similarly here in the workspace, you can see there is this variable is also there. But now x and y are sine and cosine respectively. So you know that sine and cosine from basic mathematics, that they are basically identical to each other, and they are just displaced by each other by pi by 2 more or less.

I mean the waveforms of sine and cosine look very similar. And if you displace it by pi by 2, they will actually, they can overlap. So I wanted to test this idea using the principles of cross-correlation that we had discussed. So I decided to calculate the cross-correlation of x and y. And when I do that in MATLAB, this figure pops up right, this figure right here.

So it has done the sample cross-correlation on the x axis and it has used lag, a value of lag here and this is the plot. Now there is an issue here, which is it is not plotting this, this lag is I have not given an input. So the command has automatically taken a default value, which is not to my liking.

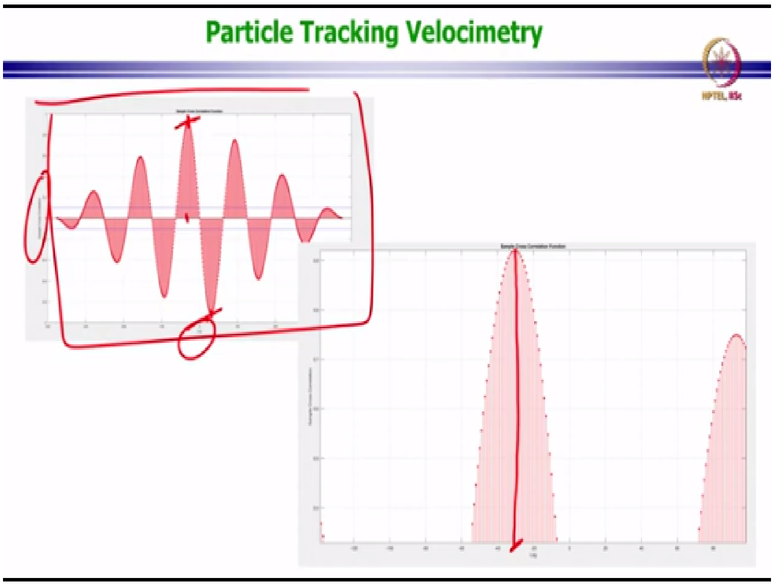
**(Refer Slide Time: 06:40)**




So I went ahead, and I looked up the documentation for cross-correlation. When I look up the documentation, it tells me that I can input my number of lags right here after x and y. And that allows me to set my lag in any way I want, okay? So you are, if you have access to MATLAB, you are free, I encourage you to read this particular documentation. This is all one dimensional calculation.

When we speak about images, we will start to look at two dimensional calculations. Still, this gives us a pretty good, the one dimensional calculations give us a good handle on what is going to happen in two dimensions, okay?

**(Refer Slide Time: 07:25)**



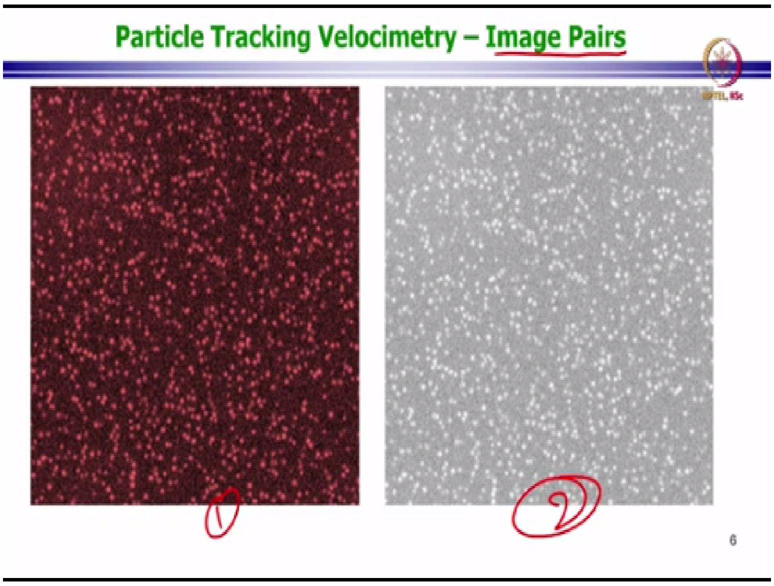

So next thing which I do is which I have not shown here. I have gone back, and okay, that command is not very visible, I have gone back and I have corrected this command of cross-correlation. And I have inputted, so I have this command x, y, 376, which is just one less than 377. And when I do that, this figure pops up right here. And you can see on the y axis, you have sample cross-correlation.

And you have lag here on the x axis, and the plot is provided to you. And you can see that at certain, there are some values of lag at which this is almost 1, the value right here. Now if you magnify this, what you find is that this maximum, which is greater than 0.9, it is, I have not calculated exactly what value this is, this happens at a value of basically pi by 2. So this lag value which it is taking in numbers of the, from the vector.

So this corresponds to your pi by 2. So you can see that the cross-correlation is telling you that at pi by 2, this is negative pi by 2. You will have another one at plus pi by 2, there is a negative value of the sample cross-correlation. You basically the cross-correlation values are telling you that if you shift your signal by pi by 2, you achieve a very high match between the two signals.

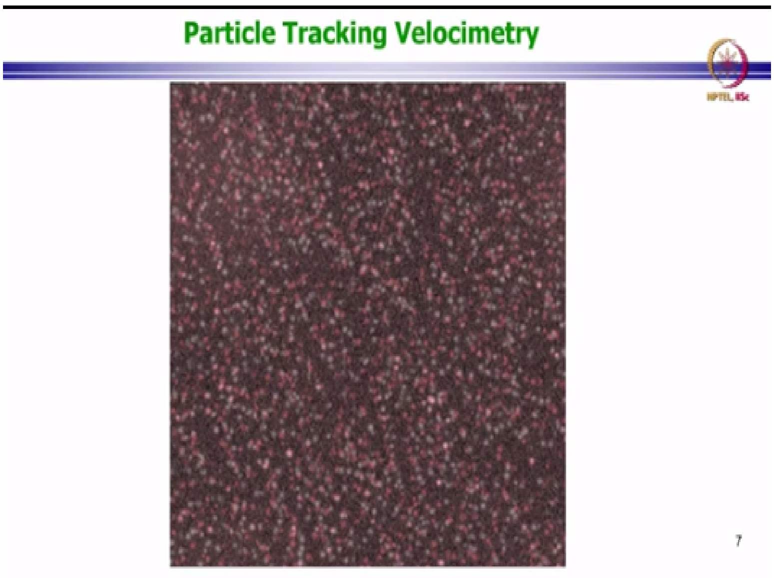
So basically, the idea was that the cross-correlation, the peak of the cross-correlation tells you where the best match comes. Now this is obviously an example in one dimensions. And when we go into images, we will start talking about two dimensional cross-correlations.

**(Refer Slide Time: 09:34)**



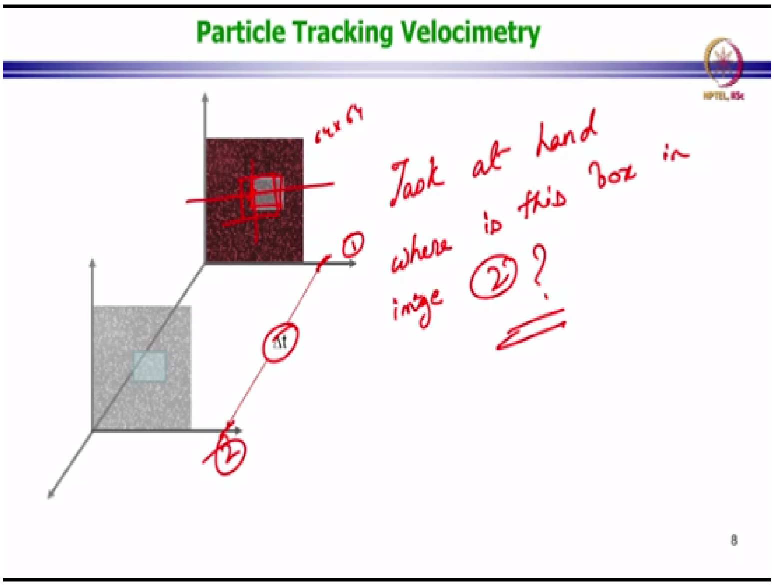

And hence, it is important for you to identify or understand how these cross-correlations work. So when it comes to particle tracking velocimetry we are going to work with image cross image pairs. And this is an image pair that we have been introduced in an earlier video, earlier lecture. And this is image 1 and this is image 2. And the whole idea is now what cross-correlations etc., we are going to do. So I am going to talk about that in a moment.

**(Refer Slide Time: 10:05)**




And I had also told you before that here, it is going to be very difficult to identify different particles, right? So the idea is no longer to do particle tracking velocimetry. You are not supposed to track particles, but you are supposed to be tracking some type of a wave form.

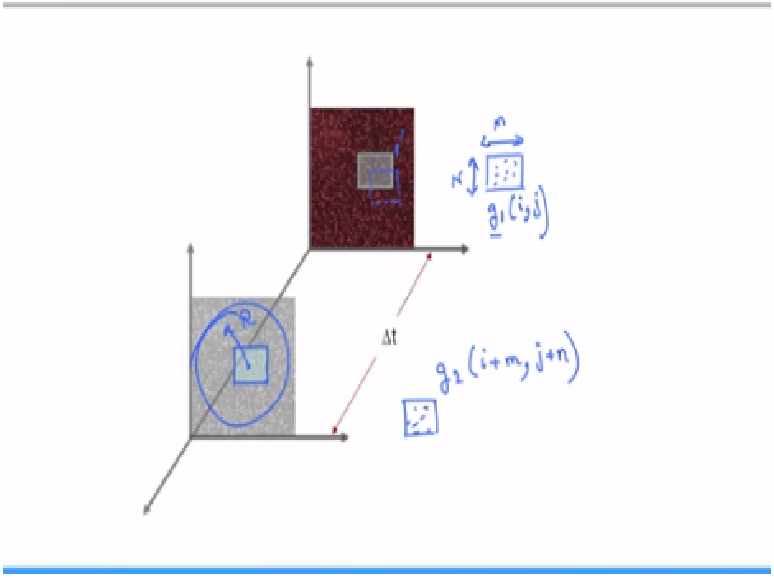
**(Refer Slide Time: 10:23)**




So in order to do that, what we do, what we are going to do is we are going to take these two images. So I am drawing a conceptual diagram. And in this conceptual diagram, in the beginning, you first your camera takes your first image, and this is your entirety is the first image. And then after a period of delta t, your camera takes a second image, and this is the entirety of the second image.

Now you select a small portion of the first image here right, and we want to know and in this small box are many tracer particles that you can see. But this small box is a small section of this entire image, not the entire image itself. And what the task at hand is where is this box in image 2, okay? So this is what we have to do. I am now going to switch to my notepad.

**(Refer Slide Time: 11:38)**



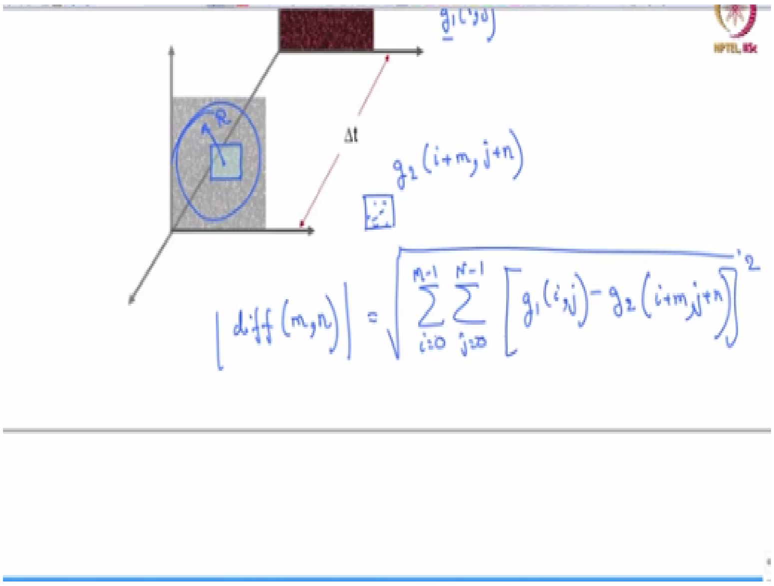

So I have the same thing, the same image here. And I am just going to draw it on the side. So basically, we have a small box okay, of let us say, m cross n pixels. And I am going to call this g 1 okay, and g 1(i, j). And this corresponds to some, this is a, by the way, these are all, we are now going to start dealing with monochrome images.

So if you are taking a color image, you must make sure that you are using a grade, a grayscale version of that, right? So this is a two dimensional matrix, not a three dimensional one. And now in this box, so now this particular small box, which we will call one dash, let us say, you have to find where it has moved in the second image.

So this collection of particles that is contained, this would have moved somewhere in the second image, and it will not be at the same location. And I am drawing this, maybe this is the new location in the second image. And you have to now look but this location is unknown. So you have to now find out where this is, this g 1 is in the second one. So let us say you have to start with some guess.

So this is maybe a guess of where g 1 has moved. So I have my g 2, which let us say is shifted in the original coordinates by m and n. And this is my guess box, okay? So I am guessing that my g 1 has now moved to the location g 2. And I have to find out which, so now what I can do in my second image is I can keep this first box. And I can design a algorithm that searches within a certain radius for this particular box, right, and it searches all over.

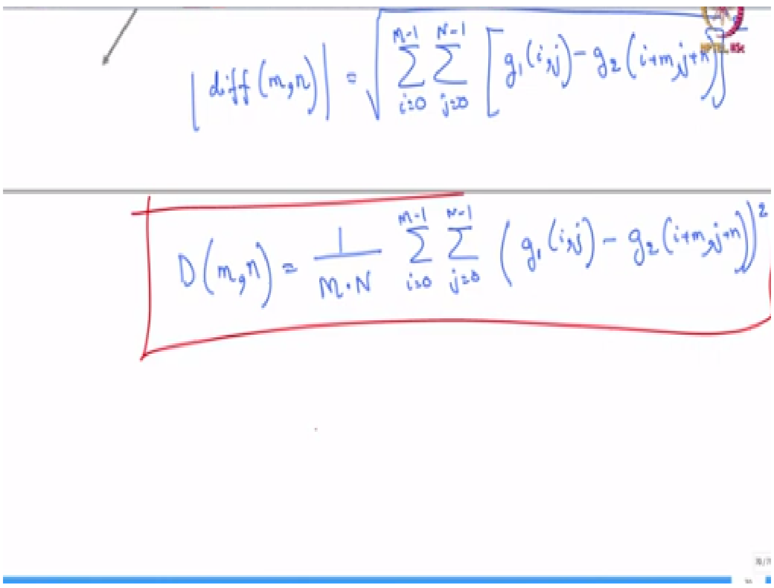
**(Refer Slide Time: 13:44)**




And one of the ways of doing this is now to calculate the difference between two images. So let us say the difference between two images in terms of m, n, where m, n is the search radius part that you are giving. And the difference between the two images can be quantified by this equation, where this is i = 0, j = 0. This is M – 1, N – 1 and here you have g 1(i, j) minus g 2(i + m, j + n) square.

Now just to clarify, this is M - 1 because my i counter starts at 0. If you are starting at 1, which is what had happened in image, this is just go to m and n, right?

**(Refer Slide Time: 14:43)**

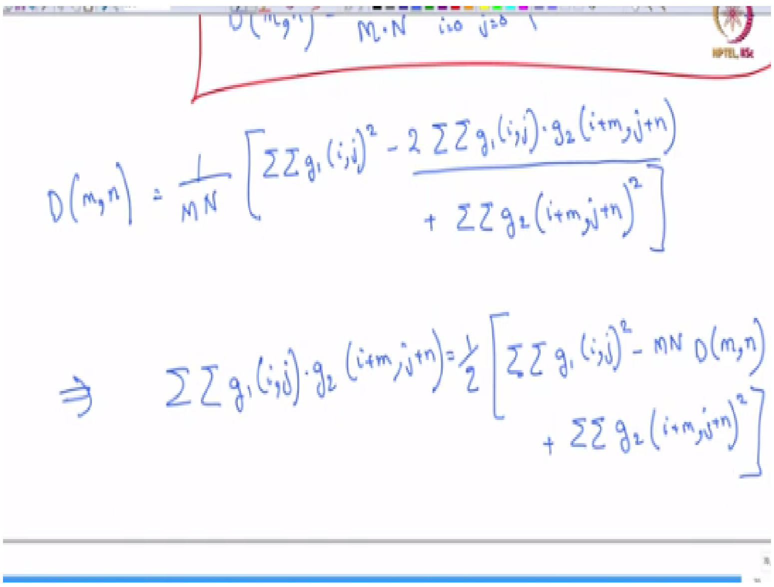



So from this above difference, you now get the idea that perhaps I can define this kind of a difference between two images, sort of a square difference, which is given by this formula g 1(i, j) – g 2(i + m, j + n) square, right? And that this difference now, it will be such that this, so this difference is basically telling you the differences between g 1 and g 2.

Now when there is an exact match, this difference will be the minimum, right? They may not be exactly, the difference may not be 0, but it will be a minimum. The reason it need not be a 0 is because you may not have the exact same match in the second image. The match may be very close, but not exact. So my D m, so this is a very important equation.

And this is also called the minimum quadratic difference method. And is this D m is often employed to search for closeness between images. Now there is another powerful tool, which is my correlation based tracking. And that uses, it is actually very much related to my D m and we will see how that is. So given this equation, we can expand this particular case. So let me do that.

**(Refer Slide Time: 16:27)**

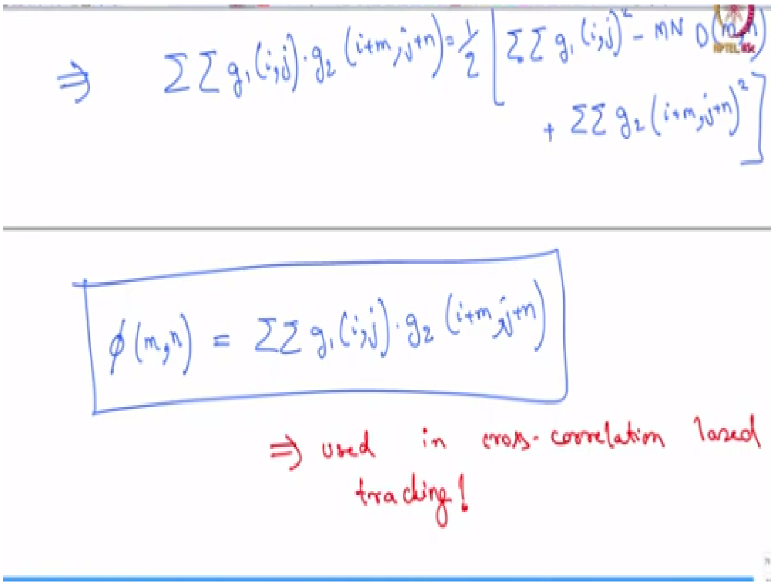



So my D (m, n), let me just go ahead and expand that. So this is MN and I am not going to keep on writing ij because it is sort of understood, so I will just put summation signs and I hope that you will realize automatically what it means. So I am just going to expand this particular case and what you end up getting is, right? Now what I am going to do is I am going to see this particular cross term that is in the middle.

I am going to try and isolate that and I will try and rewrite that in terms of everything else. So I am going to isolate this cross term and bring it onto my left hand side and this is nothing now but equal to half times summations, double summation g 1 (i, j) square minus MN D(M, n) small n plus summation, double summation again. This time it is g 2 (i + m, j + n) square.

So this cross term, in fact you can show that what it is is basically this is the, so this cross term is essentially in some ways representative of this difference plus some other terms.

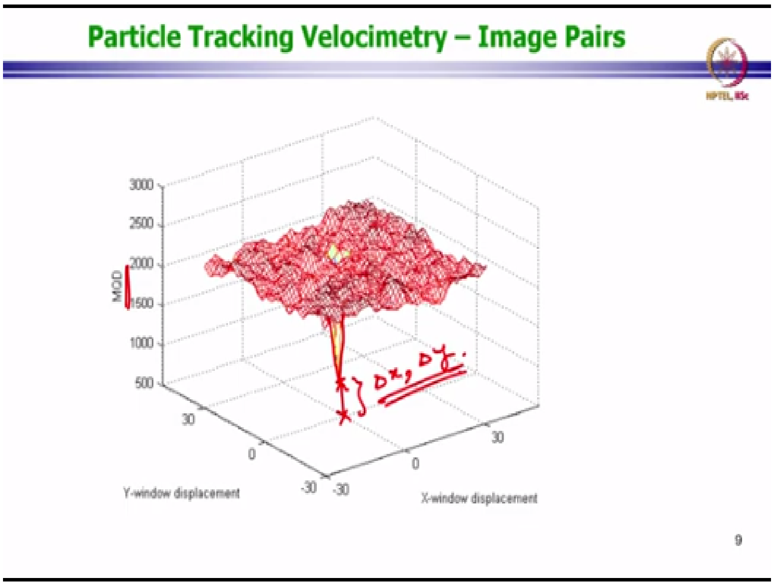
**(Refer Slide Time: 18:28)**




And this is also now often called, so often the variable phi is used for this cross term and this is now used in cross-correlation based tracking, okay? So if I were to go back to this particular problem and what I have done now is I have gone in and taken this for a particular image and then I took center location and a box at the center location of I think about 64 into 64 pixels and I searched for this box in a radius of, search radius of 30 pixels.

And I calculated my D and this phi variable in both cases and this is how the plot looks like.

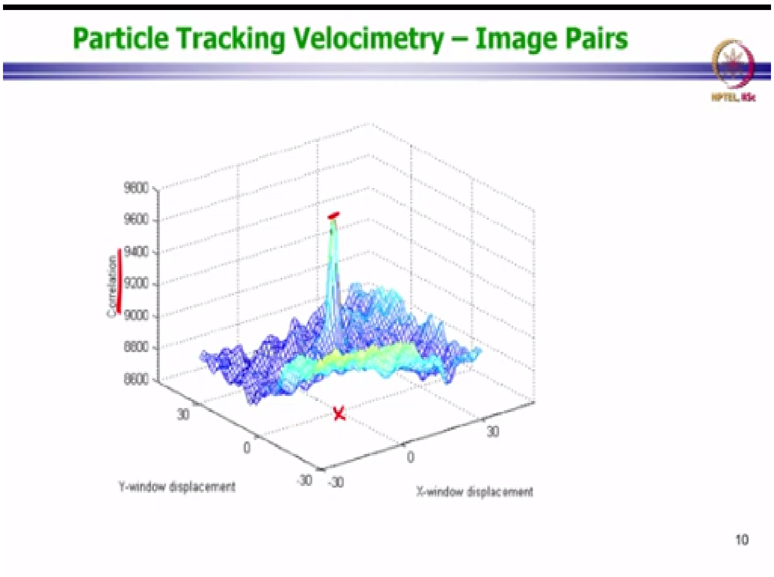
**(Refer Slide Time: 19:46)**




So this is the minimum quadratic difference plotted along x and y. And you can see that there exists a minimum right, a minimum value to this curve. And this minimum value is on my x and y the values of the displacement of this from 00 that gives me, this is not 00 by the way, you can see 00 is right here. And so this value is somewhere displaced from that.

So now you can go back and you calculate what this delta x, delta y is. And when you do that, it gives you the displacement of this box that you have been tracking in the second image. So this location tells you where this box has moved. So the delta x is that happened in delta t time.

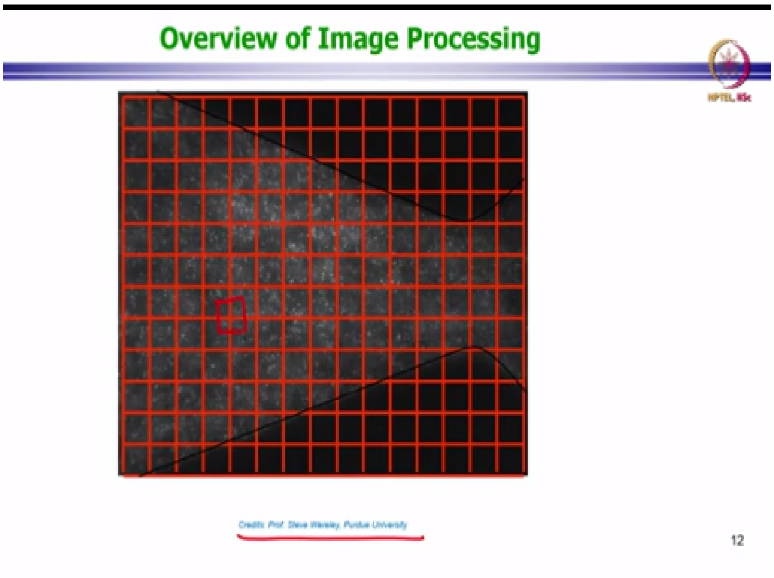
**(Refer Slide Time: 20:44)**



You can also do the correlation based tracking. And once again, you can see this 00. And here instead of a minima, you actually get a maxima in the correlation and this becomes clear when you look at the mathematical expressions, the correlation phi is a negative of D. So that is why it has a positive value in this particular case. And the maximum here corresponds to the displacement.

So you could calculate displacements in either of this ways. So you could either do an MQD calculation or a correlation calculation. And the correlation calculations are more popular nowadays. And most algorithms, they tend to use correlation based tracking. So but the two are related to each other, and you can see how they are, okay.

**(Refer Slide Time: 21:54)**

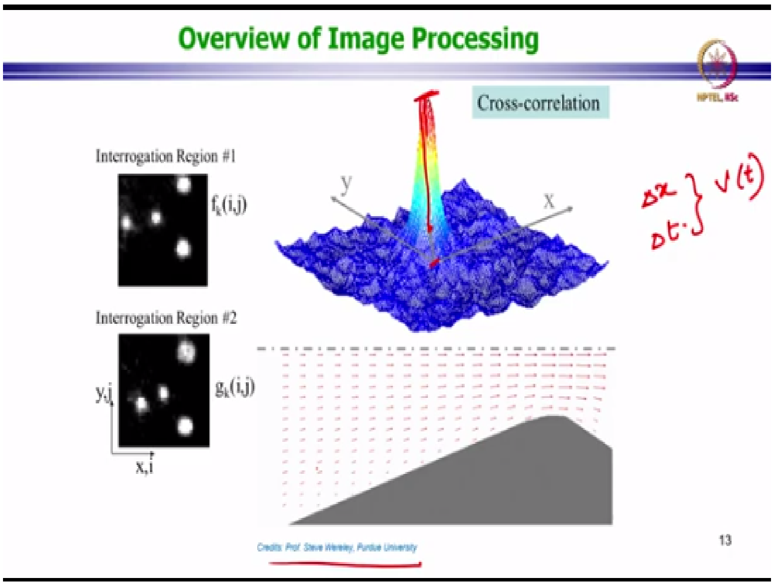



So I am just going to quickly go and try to show you now exactly what is happening in a particle image velocimetry calculation. So this is for example, let us say some two images okay, two images just put in a GIF form, and you can see the small displacements. So this entire thing is credits to my Professor Steve Wereley at Purdue, who was also who taught me, from whom I learnt quite a bit about particle image velocimetry.

So this type of an image when you get it, you when you use post processing, you will divide it into a number of grid points. And then at every grid point, you are going to do some kind of a tracking window. And this tracking window, you have to, when you once identify this tracking window, in the next image you have to figure out where the set of particles went.

So every particle window contains a set of particles, right? So it does not contain one particle. You are no longer tracking individual particles, but you are tracking a set of particles in the window. And that gives you a waveform. And what you are trying to see is where this waveform landed up in the second image. And you have the option of either doing a calculation using a MQD method or the correlation based method.

**(Refer Slide Time: 23:11)**

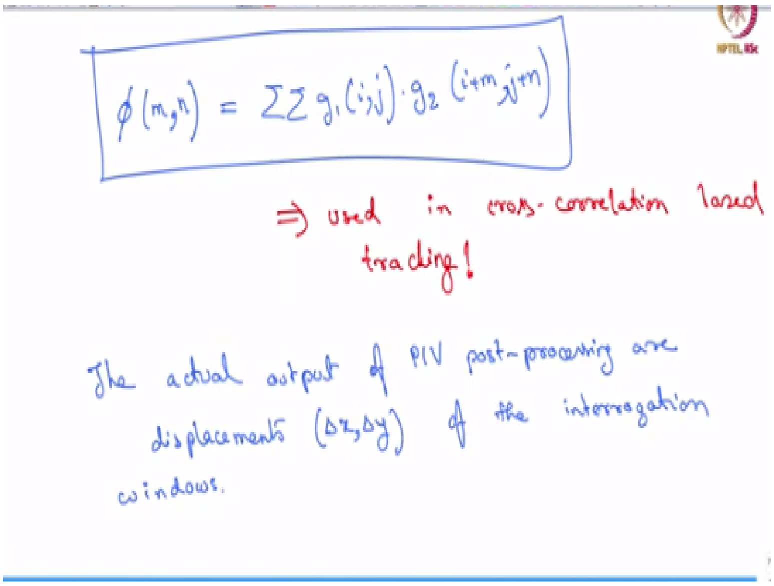



And when you are doing that, what you are going to do is you are going to do these interrogation regions. And once you have this interrogation regions, you have this small particles in the interrogation region. And then you have this is another interrogation region. This again, this slide is credit to Professor Steve Wereley and this is a very nice slide that was used in my class. So I think it is just fair enough to use that.

So when we have these two sub regions, you calculate the cross-correlation. You find out where this maxima occurs, and how far it is in the origin in x and y. So the location of the center gives you the displacement delta x. You have delta t information already, because that is something that you are controlling through your hardware and combining this you are going to get V right, so the velocity of the window itself.

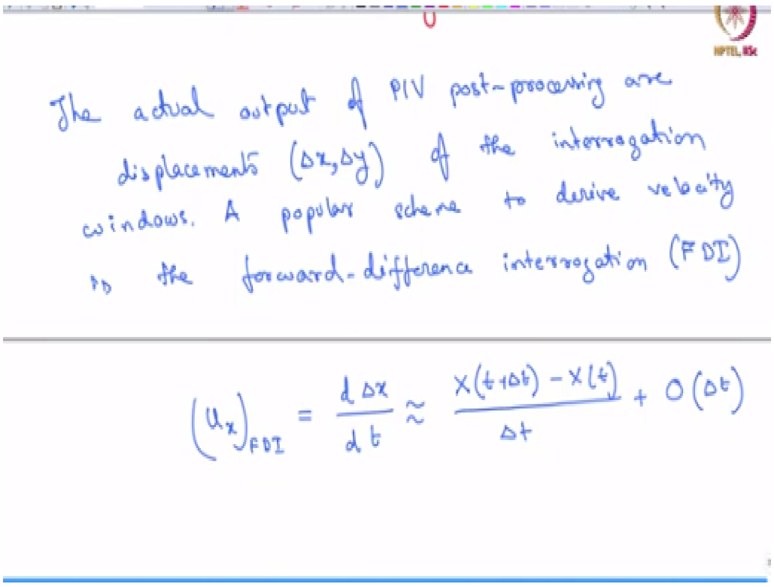
Once you have that, then you combine all the data when you are doing this interrogation for each and every single interrogation window. For example, here you do this for all the different grid points, and you end up with velocity vectors at each and every different grid point, right? Okay. So this brings us to the important question of how to find the velocity, right? So we know how to find delta x. Now the question is how to find the velocity.

**(Refer Slide Time: 24:46)**




So the actual output of PIV post-processing are displacements, which means my delta x and delta y of the interrogation windows and from this you have to calculate the velocity, right?

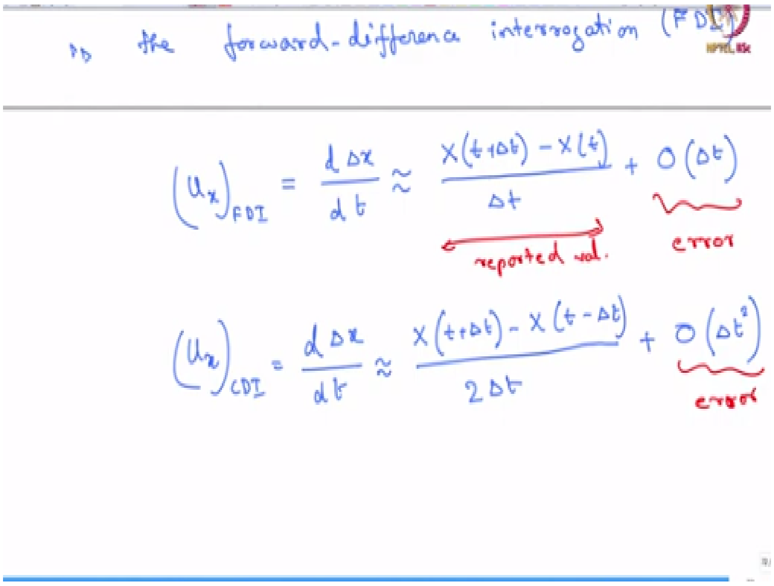
**(Refer Slide Time: 25:25)**




So a popular scheme to derive velocity is the forward, sorry forward-difference interrogation, also called FDI. And in this what you do is use the forward difference idea from numerical analysis to calculate the velocity. And the idea is that you want to calculate this quantity right dx dt or d delta x dt.

And this you approximate by calculating this quantity which is x t plus delta t minus x at time t delta t plus order of delta t terms. There will be other terms which will be of the order of delta t.

**(Refer Slide Time: 26:46)**

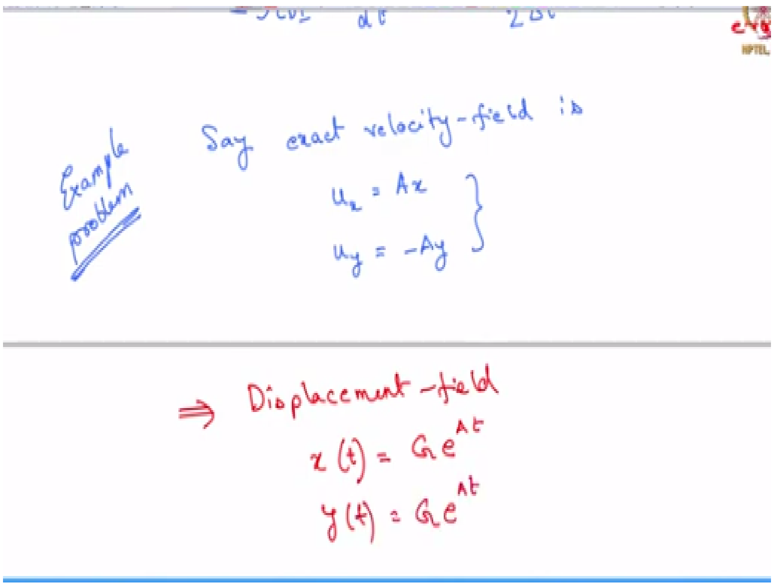



Now when long exposures are used, when long exposures are used the errors in the finite difference interrogation scheme can be very large in which case the central difference schemes are preferred. In central difference, you calculate, try to calculate the same quantity, but you apply a slightly different formula for that. And what you do is you do a t plus delta t here minus x at t minus delta t and this is divided by 2 delta t.

In this case the order and you can show using Taylor series approximations that the order now of the error is reduced. So this is the error by the way obviously the error and this is your reported value. And the order of error in the central difference scheme is of the order of t square. So as you decrease t again and again further and further, you end up getting a better result in the central difference case.

We want to see this with an example. So what I have done is I have composed a example problem.

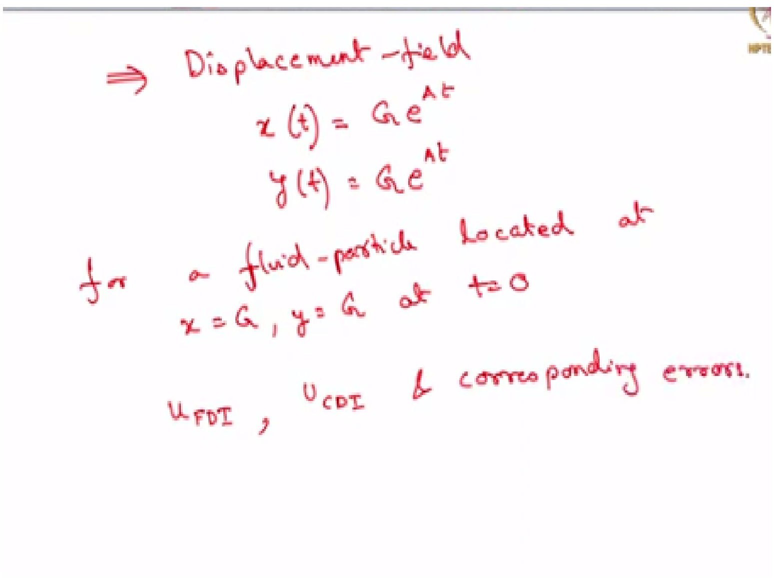
**(Refer Slide Time: 28:05)**




So let us do this example problem. And let us say the exact velocity is available to you, okay? So I say exact velocity field is available and this velocity is given by A x U x and for U y it is minus A times y. And you have to calculate the error in evaluating this using the finite difference and the central schemes. So now in order to do that, we have to first calculate the displacement errors, sorry the displacements by the way.

So we know that when the velocity field is given the displacement field is also automatically known and that you can get by integrating the equation above. So my x t is nothing but some G e to the power A t and my y is again some G e to the power A t, right?

**(Refer Slide Time: 29:19)**



For a fluid particle located at x = G and y = G at t = 0, okay? So the task at hand now is to find what U FDI is and U CDI is and corresponding errors. So please apply the formulas that we have discussed here and please calculate in this particular case, the errors. And we want to understand that these errors are of the order of delta t and delta t square and how exactly so this example will help us understand that better.

So I will pause here for a minute, and I will allow you to try and do this. And I will pick up back from this particular place shortly. So thank you.