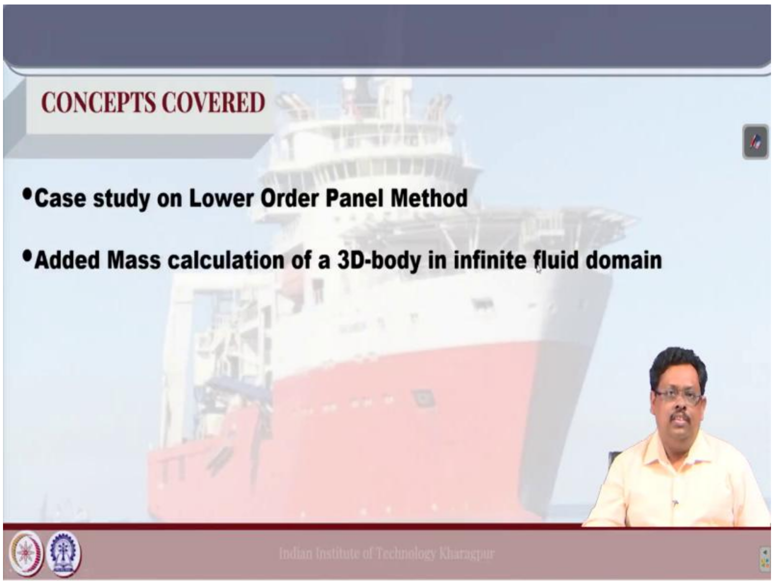
Numerical Ship and Offshore Hydrodynamics
Prof. Ranadev Datta
Department of Ocean Engineering and Naval Architecture
Indian Institute of Technology, Kharagpur

Lecture - 19
Case Study Part 2

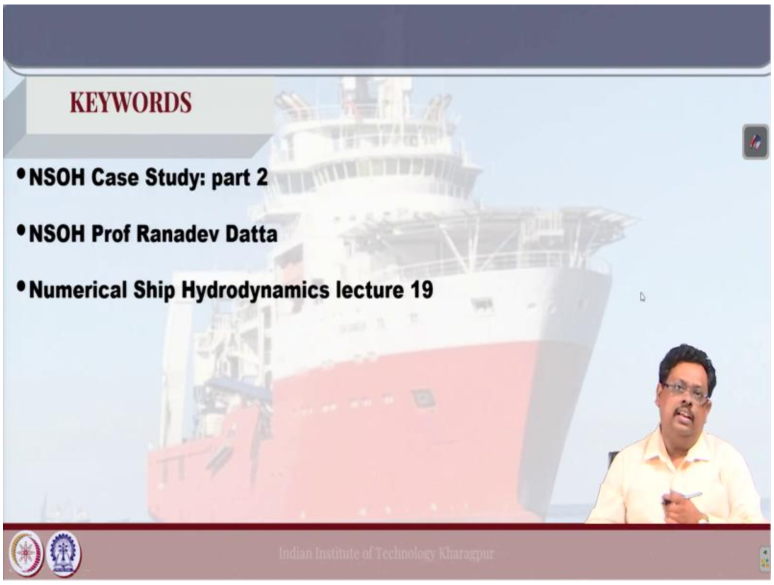
Hello welcome to Numerical Ship and Offshore Hydrodynamics.

(Refer Slide Time: 00:17)



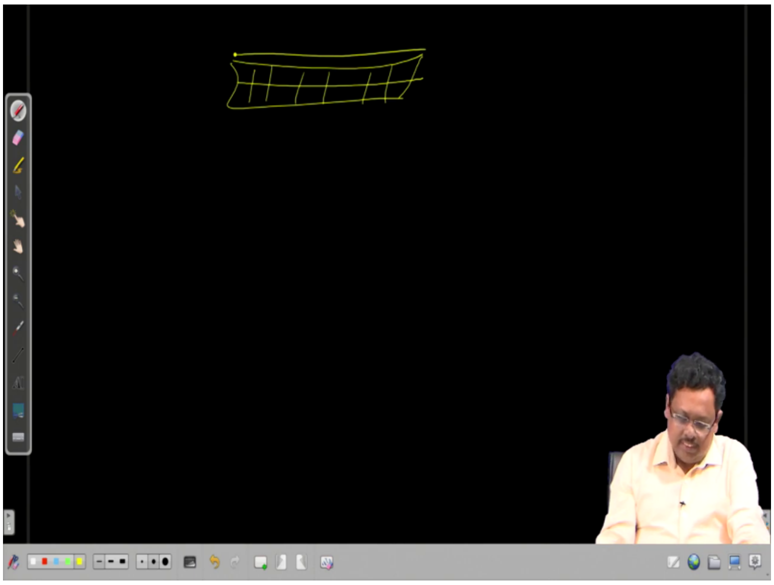
Today we are going to discuss that we continue from the last class to take the case study, to get how to find out the problem of the infinite fluid domain to get the added mass and damping etcetera. You do not get the damping to get the added mass ok.

(Refer Slide Time: 00:39)



So, this is the keyword that we are going to use to get this lecture ok. So, now let us continue from where actually we stopped in the last class. So, in the last class we tell that you know to write a good programming and in order to get the input very correctly, you have to do lot of initial things, and you have to check lot of initial things.

(Refer Slide Time: 01:19)



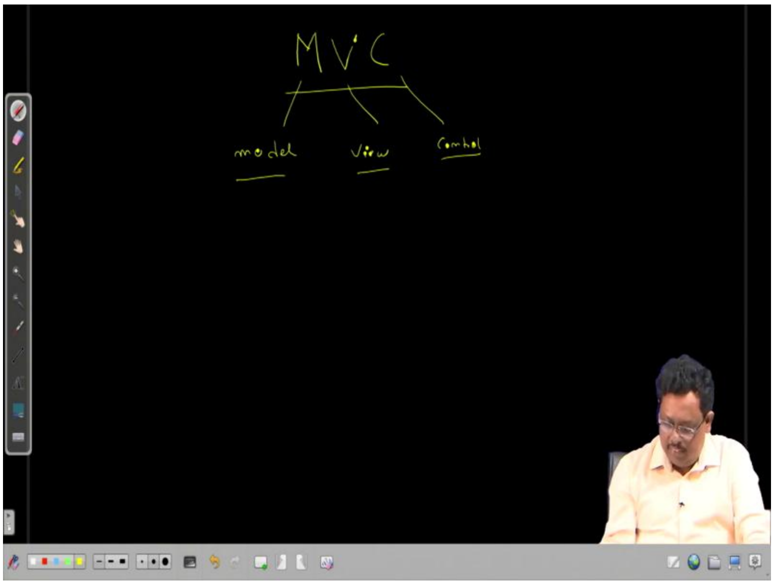
For example, if you remember. We said that when you do the meshing etcetera. So, this is the ship and when you do the meshing etcetera so your input is the meshing. So, therefore, you need to do the quality checking of the mesh before you further proceed right.

And then we have lot of discussion what can go wrong and then you know what you should avoid, what you check first everything we have discussed in the last class. Now, today we really start to develop a code and how to write a good efficient code let us try to discuss today. Now, if you look at this particular problem to find out the added mass in infinite fluid domain, using the panel method whether g = 1/ r. So, these are the physical problem we are going to solve.

Now, in order to attack this physical problem, in order to write a good code you have to know remember certain basic things, what are those? First you know this is some we normally let us set this lecture with respect to some object oriented coding language right. And also that MATLAB also you know you can follow this the same way. However, normally it is seen that if you try to do a commercial coding. So, normally people use C^{++} or C sharp something like this or visual basics all are basically objective in a programming language right.

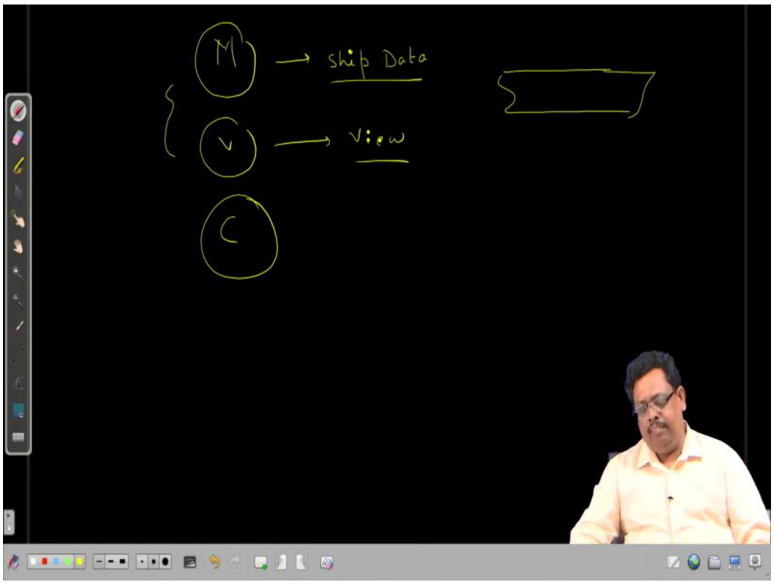
So, and also this type of code would require lot of playing with the data, lot of play with the geometry to set some detail logic it is always beneficial to use some language which actually build on some structured language I would say, like a C sharp or C^{++}.

(Refer Slide Time: 03:34)



Normally, we if we try to write a model for this type of thing, we can call this a MVC model. Now, M itself is a model and V itself is a view and this is controller or control. Now, what do you mean by model? What you meaning by the view? What is the meaning by the control? Let us try to explain one by one. Now, model means that for your for our case the model is my the ship, the ship data and the physical problem that is my model.

(Refer Slide Time: 04:18)

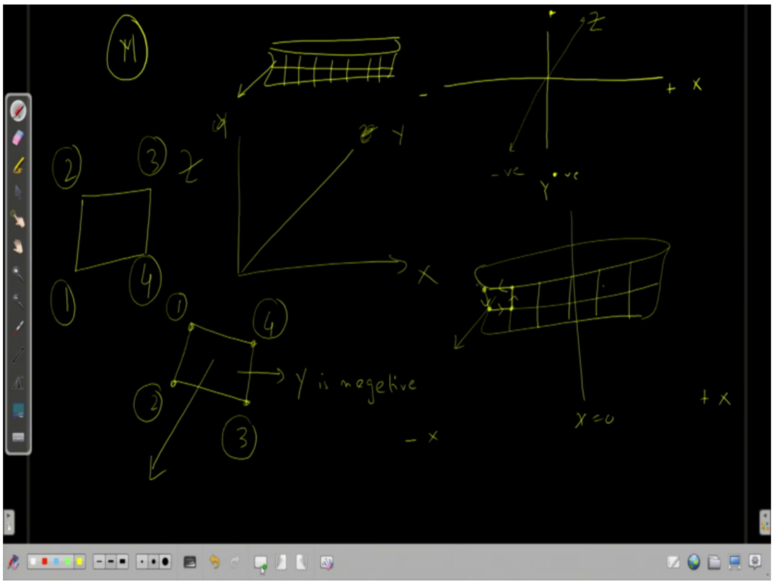


And then; so in M is transform that my ship data right. How I write or how I store my ship data. So, this is the first thing and then how I can view the result right. Now, how we can high; I mean it is actually we can see lot of way you can view our result. Somebody wants some graphical way to see that, like I give the ship data and then I need to you know you know see my the whole ship is coming in front of me.

So, somebody can say ok, this is how I can view the data. Somebody wants ok you know let us if the you know user wants then I can give back their data ok, that means in a mathematical form or in terms of some excel file or some tabular form I can give the data. So, then one aspect is like how we can show to the user ok, the end user. And then this through actually we control through the point controller or we can call it C ok.

So, now let us try to for this particular problem, let us start from the basic that for M, how I keep the model and then how I control it and finally, how I can view this. So, let us do step by step ok. Now, the first part that how I write the data that we discussed in the last class. Now, let us now try to solve the problem using that data. So, at end of everything what we have right now is basically the elements or the panels.

(Refer Slide Time: 06:22)



If you remember what I said in the last class, suppose I have this ship. So, the final we stock at this point that we have some panel or triangular or quadrilateral mesh. And also, we understand that we have to find out if the normal are outward drawn normal or inward drawn normal.

Now, how to make sure the data should be you know uniformly the normal should be uniform outward drawn or inward drawn, that we are going to discuss now. So, let us first try to solve that M part, that is in the model part or data part how we are going to take it.

Now, see here if you look at this coordinate system. Now, here X is positive and then here the X is negative. Now, let us take the transverse axis; that means the Y axis. So, let us this is your X axis, now this is your Y axis. So, this side is positive, let us say this side is negative ok and then you have the Z axis.

Now, in Z axis the upward is positive and then this downwards is negative ok. So, now, I have now understand then I have let us say draw this three system, this side X is positive, this side Y is positive or and this side the Z is positive. So, let us say Z of vertical side ok. So, let us change its, let call it Z and let us call it Y.

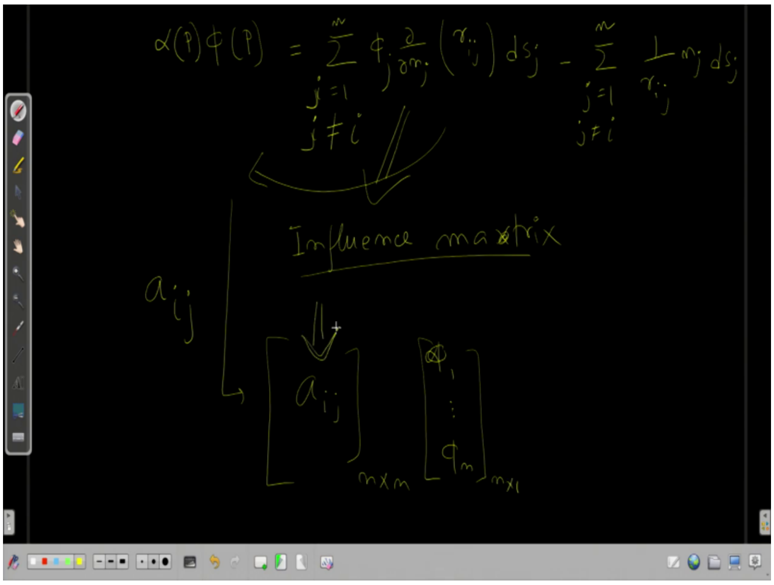
So, now, if you draw the ship and let us make sure that, I need that outward draw normal should be inward to the fluid, then what I need to do is; now if this is my panelling and if you take the middle part is X = 0, this side is let us say plus X and this side is minus X. Then and then I should take the meshing should be in this form.

So, I said that my meshing should be here anticlockwise. So, if it is the point number 1, it should be the point number 2, this should be the point number 3 and this should be the point number 4. So, if you do that, then it is always the normal should be the outward to the body inward to the fluid.

Now, in other side in other side so, now, if this side I try to make. So, this is the side when y is negative. Now, when here this side is Y is negative ok. Now, suppose where the Y is positive; that means, the other side of the vessel how I can you know orient the number 1, 2, 3, 4. Now, in that case you have to go clockwise. So, it should be 1, then it should be 2, this should be 3 and this should be 4, get it.

So, now in this side our Y is negative I must go in antic clockwise direction to get the normal you know outward to the body, inward to the fluid. In other situation, I should go in the clockwise to get the normal outward to the you know body inward to the fluid. So, this is how I should do my meshing. Now, this part is done ok. Now, let us see that how I can write the formal code.

(Refer Slide Time: 10:48)



Now, you see; now if you write the equation in summation form, it should be $\alpha(P).\phi(P)=\sum_{i=1}^{n}$ and then i, ok let us take j j = 1 to n and then j $\neq$ i right and then first part is ϕ_j and then $\frac{\partial}{\partial n_j}(r_{ij})$. So, this is the first part into ds $_j$.

So, this is the first part of the thing and then, it is minus sum over j = 1 to n and then j $\neq$ i and here it is $\frac{1}{r_{ij}} n_j.ds_j$.

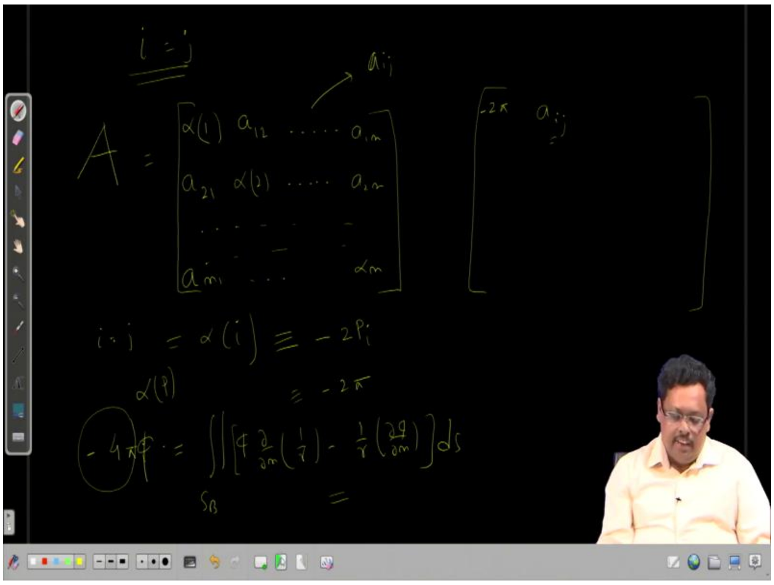
$$\alpha(p)\phi(p)=\sum_{j=1, j\neq i}^{n} \phi_j \frac{\partial}{\partial n_j}(r_{ij}) ds_j - \sum_{j=1, j\neq i}^{n} \frac{1}{r_{ij}} n_j ds_j$$

So, this is so you have now two things, right. Now, this part we call as influence matrix. So, in this influence matrix what are the terms? So, I call this influence matrix as a $_{ij}$.

Now, if you remember, I said i $\neq$ j. Now, what is happening this influence matrix should come in the left-hand side and then this will at the end I will get a square matrix which is a $_{ij}$ which is of course, n cross n matrix and with this I have the matrix for ϕ which is ϕ_1 etcetera ϕ_n. So, this also should be (n x 1) matrix, right.

So, therefore, the second part I need to get this influence matrix right. Then how I could get this influence matrix?

(Refer Slide Time: 13:12)



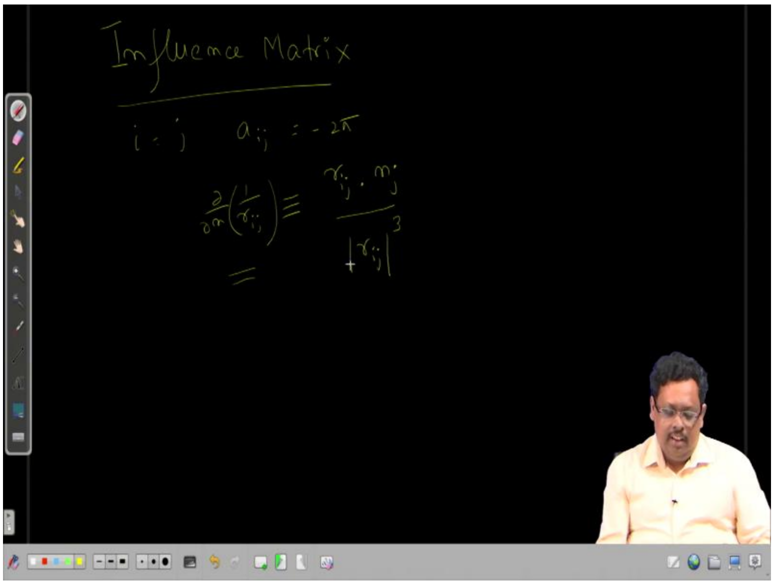
Remember I said when i = j, then we do not have to do the integration. So, then this is how I this matrix let us let us call it A. So, this matrix is look like this. So, the first point for the ϕ_1, it should be the α_1, I call this as a α_1 because for i = 1. And then it is a 1, 2 and in this way it will go a_{1n}, then it is a_2, 1 and I call this as $\alpha(2)$ and then it is a_{2n} and in that way it is going and finally, it is a_{n1} and then it is $\alpha(\mathrm{n})$. So, this is how I should write my a_{ij} matrix.

Now, when i = j, it is called is ϕ_i and now this α in our case we should take this as - 2π ok. So, how I get this -2π? We are definitely going to discuss this, but if you remember in fact, in our previous lecture if you remember, that we have 4π ϕ is equal to, I have this integral equation which called this over the S body and it is $\left[\phi\frac{\partial}{\partial n}\left(\frac{1}{r}\right)-\left(\frac{1}{r}\right)\left(\frac{\partial\phi}{\partial n}\right)\right]ds$.

So, from this equation actually, I and then I replace this by solid angle which is α. Now, when i = j this value of α becomes -2π half. So, when we are going to discuss this later on, when we actually do this 1 me type solution or the or the panel method for frequency domain panel method for floating bodies, we are going to discuss that part that time we come to know how this value comes as -2π. But for the time being let us take this as I said when i = j the matrix entry should be -2π.

So, therefore, this entry now becomes always, so when i = j the value should be -2π ok. And then, when i $\neq$ j, then I have to do this a $_{ij}$ part, right.

(Refer Slide Time: 15:57)

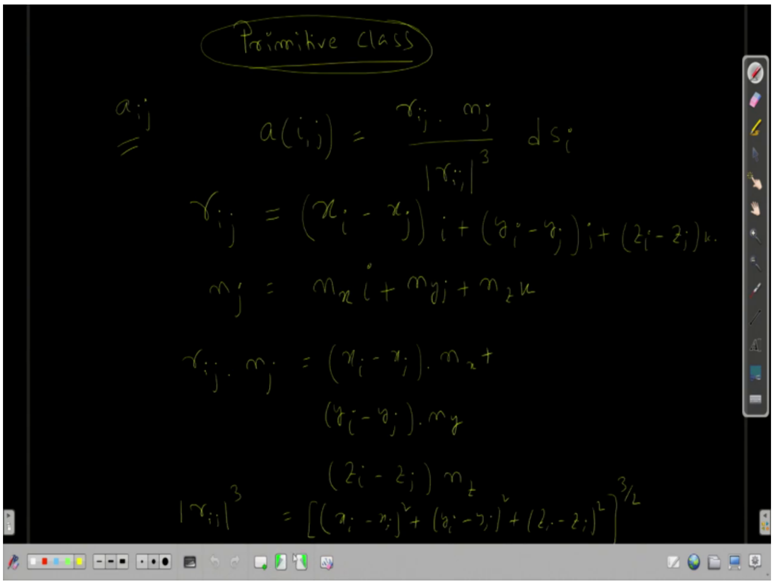


So, now, I understand, now I first I need to write my influence matrix. Now, I am now I now, I understand that when i equal to j, thus entry a $_{ij}$ should be is equal to -2π. And when i $\neq$ j, that time the my entry should be as you know it is $\frac{\partial}{\partial n}\left(\frac{1}{r_{ij}}\right)$. And if you remember we have solved it this is nothing but $\frac{r_{ij}.n_j}{\left|r_{ij}\right|^3}$ right, that we have already discussed.

Now, see now my now this actually tell me that how to write the influence matrix. Now, here comes the coding. Now, we are very we know that how to get this the get this value of this, but how to arrange it in a code so that you know when. Now, thing is that you write a code and then maybe somebody wants to develop the code based on that what you have written. But then he should understand your code. So, your code should be logical right and it should be understandable to the other person right.

So, in this case that how actually we proceed? Now, you see that if you do in the object oriented programming. So, now, in order to write this code very efficiently, now in object oriented programming language some cons it is called the primitive class.

(Refer Slide Time: 18:00)



Now, what is primitive class? Now, this is the class it is something called the most basic class, where you your input is minimum and then output is minimum.

Now, what when you say that input is minimum and what you call the output is minimum, what is the meaning of that? Now, you see suppose you want to perform this equation to get this a (i,j) this coefficient, then you have to do at some point some a(i,j) is you need to do this operation, which is $\frac{r_{ij} n_j}{|r_{ij}|^3} ds_i$. And of course, you need to multiply with this the area associated with this right.

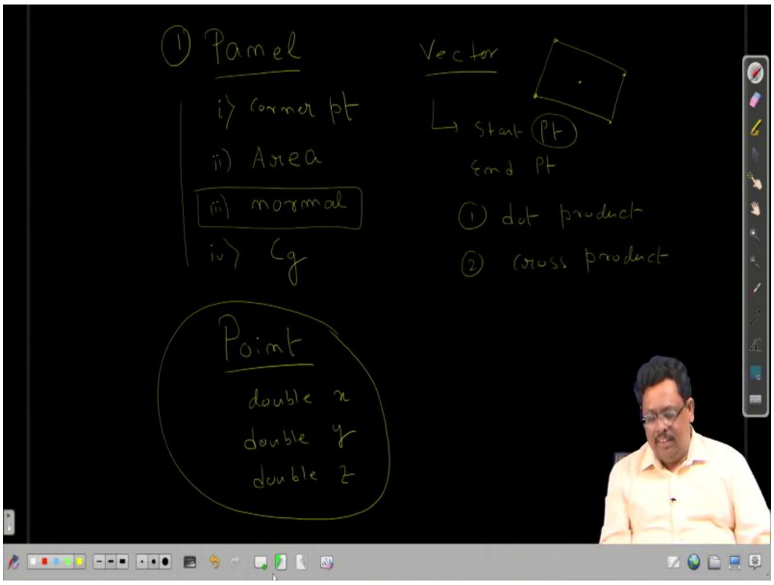
So, this you need to do, you see it is not a single task. Now, if you break it stepwise, then what is the r_{ij} it is the distance between the source point and the field point.

So, this nothing but your $(x_i - x_j)_i + (y_i - y_j) + (z_i - z_j)_k$. This is something you have to calculate this r_{ij} first and then you need to calculate the second thing which is the normal, right. So, normal also have some three component $n_{xi} + n_{yj} + n_{zk}$ and then you have to do the dot product. So, dot product is $r_{ij}.n_j$ which is nothing but $(x_i - x_j).n_x + (y_i - y_j)n_y + (z_i - z_y)n_z$.

So, many operation you are doing and then finally, you have to do this $|r_{ij}|^3$ which is nothing but it is $\left[(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2\right]^{\frac{3}{1}}$. So, many things; so many things you need to do. Now, if you write everything inside a single code, then you are in trouble. Because, when you do all such thing at the end if you get something wrong, your program is not running correctly then you really do not know at which steps actually you are making mistakes.

So, in respect to avoid this, we have to write some primitive class. Now, here how do I write the primitive class here?

(Refer Slide Time: 21:38)



I could write some primitive class. So, first I can write one class which is called panel ok and then in this panel actually I can define some property of a panel. Now, how do I define a property of a panel? So, one panel is you know that it should be the coordinate of the 4 point. So, the property number 1, inside this I can put the you know the corner point. So, this is number 1.

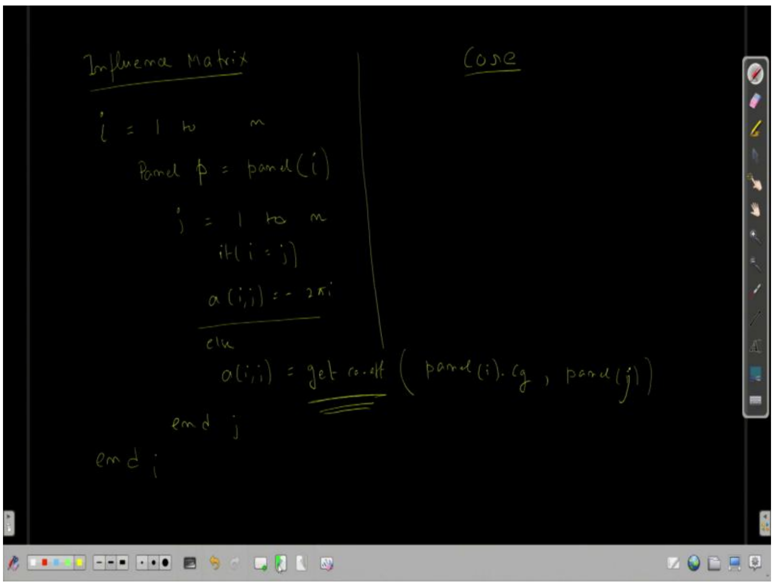
Second one this also I want to find out the area. So, second property I can find out the area right and then the third property I can find out the normal of the panel. Outward the normal to the panel. So, this is the three property actually and also fourth one of course, I forgot which is the center of gravity or we can call as Cg. So, you see that this 4 is essential for a panel. So, I define this. Now, here this normal is also not primitive.

Now, this is normal is nothing but a vector. So, I can take another class which is called the vector, then in this vector class you can have that in order to draw the vector you need a start point and the end point. So, in this vector you can have something called the start point and something called end point right. And also, once you draw the vector so you have to write some primitive some function, some normal primitive function. What are the function? For example, dot product because you have to do the dot product if you remember. So, I am writing dot product, I need to find out the area.

So, I need to write the, a function this is called the primitive function, which is called the cross product right fine. So, you see that in I just make one class vector, but it again requires some points. So, I need to write another primitive function or another primitive class we can call as point. In this point what you have? In this point you can have some function which is double, double x double y and double z. Now, this point actually called the most primitive class, which only having the basic thing the double integer.

So, now any class that you are dealing with, the double and integer it is called the primitive ok. Now, you see how we can break this influence matrix in using these primitive classes. So, first thing you need you have to write and this logic.

(Refer Slide Time: 25:12)



You see you have the two part, one is called the core which based on your geometry or doing the integration you need doing the differentiation all these things is the core engine class. Another is called the business logic, where actually you are writing the basic logic of the program.

Now, let us see how I can write this influence matrix with the help of these three primitive class I would say. Now, see again in my business logic when you call this it is my how I write it. Now, suppose this class is you know we can you can again call a function. So, you can call this a influence matrix.

Now, in this influence matrix you can write this let us say i equal to let us say some 1 to n, something like this i = 1 to n. I am just writing the business logic, say it should be end i and then you are writing another and then when you see this then you have to, now I have used introduce my panel right.

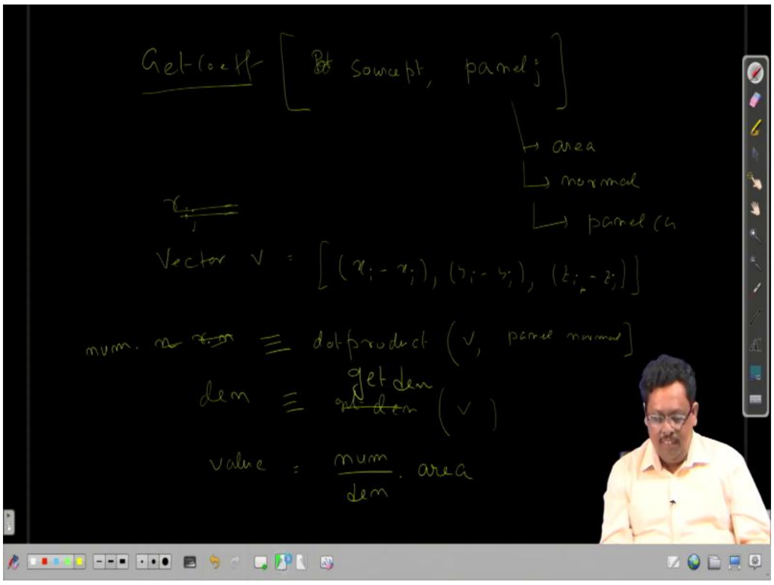
So, then I can call some panel p is nothing but I am yeah I am taking it is nothing but my from my panel list. So, I have the list of panel which is your mesh. So, I am taking the panel(i), i^{th} number of panel ok. So, now, again my logic says j = 1 to n and we are using here the end of j and then your business logic starts over here if i = j. So, your matrix entry a (i , j) = -2π and else.

Now, here this is the thing else then you call a(i , j) and you can write this equal to some function let us we can call it get coefficient and then you are sending your panel(i) which is the source point basically. So, you can finding out the panel(i) and source point basically your the centroid.

So, I am sending the panel i and then actually I am sending the panel j ok. So, panel make it panel j. Now, you see this is your business logic end. So, you see here now if you read this program anybody if he reads this program he understand very easily, that when i = j I am getting $-2\pi i$ something and when i $\neq$ j, I am calling a function which gives me that coefficient.

So, here I am not doing anything all this core logic which actually involved here, this I am not doing here right, I am not doing here. So, what I am doing here I am writing the business logic. So, then now my job is to write the get coefficient a function separately.

(Refer Slide Time: 29:07)



Now, when you write this 'get coefficient function' separately, there is also you need to break this function that get coeff. So, this function also now I just break it here. Here I can get it the 2.1 is some point which is my, the source point right, I can call this is a source point and also I am sending a $panel_j$.

So, this is if you remember my base class. So, base class have this panel j. Now, what is the information this panel j is having? This panel is having the coordinate, this panel is having the area of this panel, this panel is having the normal of this panel, that information this is having and also it has panel Cg. All this information this panel is having.

Now, therefore, in get coefficient here your work becomes very easy. So, now, you are define your r_{ij} and actually here also you can in fact, better here if you actually introduce some vector. So, you can use some vector because you have defined your class vector also.

So, you can introduce a vector V equal to and this three component is nothing but $(x_i - x_j),(y_i - y_j),(z_i - z_j)$, right. Now, you see everything is now very easy. Now, in here the running vector logic is really easy. Now, this r dot n, this $r.n$ now is a simple one function already I have written in my primitive vector class, which is nothing but the dot product.

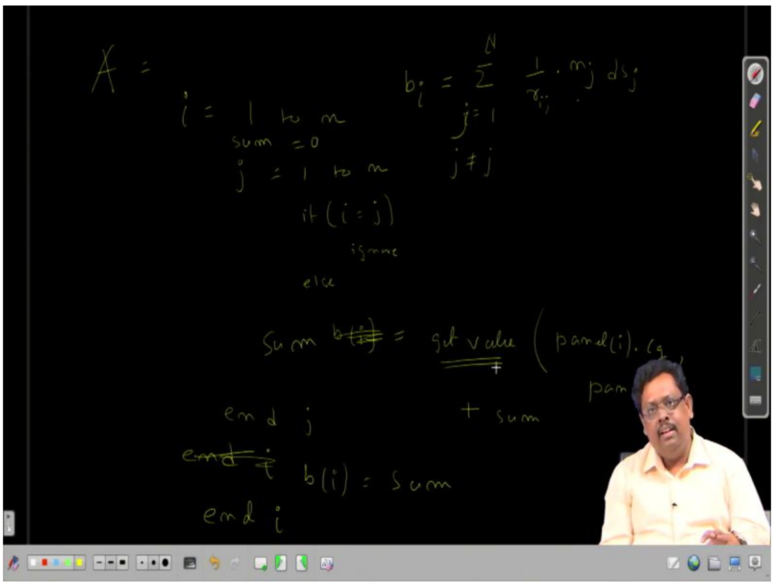
So, you write dot product and then you can call it is you are sending the vector V and also you are sending the panel normal. And also, here you can call is the - is a numerator or you can call is n and again or num whatever that base whatever variable you can take is a dummy variable or denominator also you can call as. Now, again you can write a function right you can write that get denominator.

And then you are sending your vector V, that is it. So, when you send the vector V and then you are writing a separate function for the denominator, which is get denominator here ok. I think let me write in little bit bigger way. So, just because these are all are small functions.

So, in denominator you are writing get denominator. So, you are getting the denominator and then finally, you can write your value is nothing but your numerator divided by your denominator and then multiplied by the area. This panel area is always with you. So, you can multiply by panel area right.

So, you are having your get coefficient, again you coming back over here, see in this get coefficient you are getting the value of a_i, see. Now, see I am just taking this influence matrix to show you how we can write the matrix a (i,j) or influence matrix. So, you see that in your case now again.

(Refer Slide Time: 32:50)



So, in your case now your a (i,j) matrix or a matrix. Now, you know how to write it. So, in similar way try to think how we can write the right hand side matrix which is nothing but your this $b_i = \sum_{j=1, j \neq i}^{N} \frac{1}{r_{ij}} n_j ds_j$ right?.

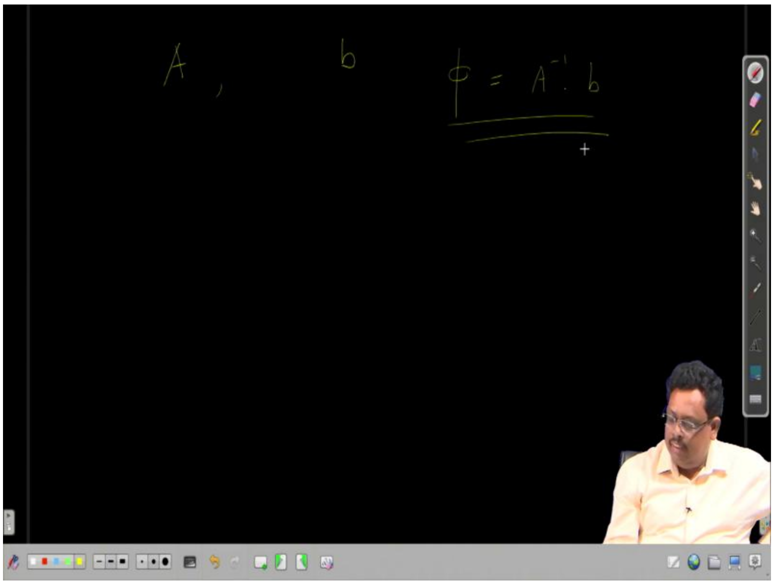
Similar way you need to write the right hand side matrix also. Here also you break this thing in small small function. Now, you see here also you have this what you need over here, see here also you can write if you write the business logic here which is again i = 1 to n right. And then it is end of i.

Similarly, you have j = 1 to n and you have end of j. Now, here your logic if i = j you have to ignore right. So, we have to ignore it and else if $i \neq j$ then you set your b(i) again write a function which is get value something like this, here you can send your that (panel (i). C_g) So, the center of the panel (i) right and also you need to and also you need to send the panel (j). And then this plus your well, we can call this as because you have to it is a summation.

So, you have to say sum equal to this value plus sum and here each time you make your sum equal to 0 right?. So, because you when you enter here the sum = 0 and then it is for all j, it will get a value and before it is end of i you can call your b_i equal to sum and then you can write your end of i. So, here also you are writing the get value. Now, get value you write a separate function which tells you this multiplication between $\frac{1}{r_{ij}} n_j ds_j$ that parts you do there.

So, now with this you that whole that coding you are actually split into small small small function, with respect to some primitive classes like points like a vector, like panel with help of this small small classes you can actually write a very good program.

(Refer Slide Time: 36:10)



Now, once this is ready, once it is ready you have your influence matrix A, you have your the right hand side matrix the small b. So, then you can just a single line the $\phi = A^{-1}.(b)$ gives you the value for ϕ right.

So, now once you have the ϕ then you are ready to get the value for added mass ok. So, we are going to stop here. So, in the next class, what I am going to show you I am going to show you a object oriented programming where actually I make all these small small classes, I will show you how I arrange this class and then how I do the business logic. And finally, how I get the results. So, these things we are going to discuss in the next class ok. Till then.

Thank you.