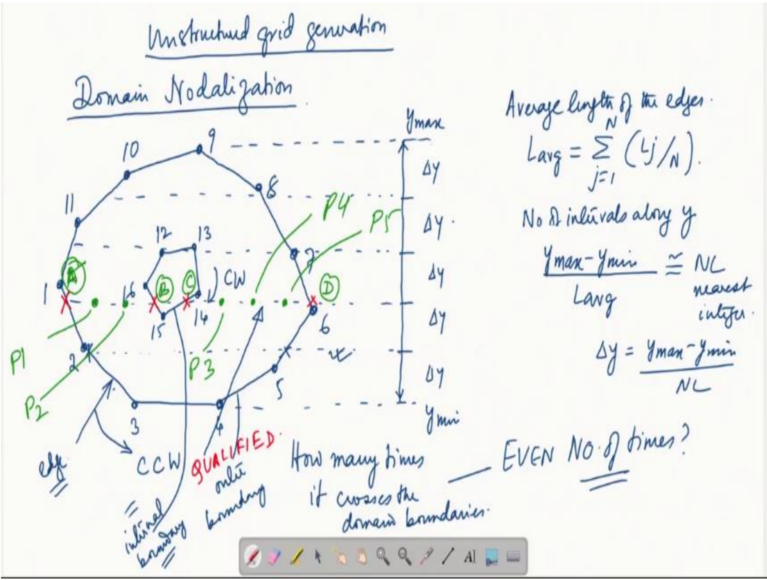
Introduction to CFD
Prof. Arnab Roy
Department of Aerospace Engineering
Indian Institute of Technology - Kharagpur

Lecture – 65
Structured and Unstructured Grid Generation - Continued

(Refer Slide Time: 00:38)



This is the last lecture in our course where we are going to discuss about unstructured grid generation in little more detail. We will try to understand some simple techniques by means of which we can attempt unstructured grid generation ourselves. So the first step in unstructured grid generation would be what is called as domain nodalization. That means creating nodes within the domain, how do we go about it?

So far doing that, let us first draw a simple domain so that we can explain the exercise, the process. As you can understand, these are edges of the domain and these are nodal points, simply nodes, and you have an inner surface also with nodes. So the outer boundary nodes would be marked in a counterclockwise manner. So we have say 11 nodes in the outer boundary and you would prefer marking the inner boundary with clockwise numbering.

We can continue with the same numbering, so the last one was 11 in the outer boundary, so here it will be starting with 12. So we have marked total nodes. Now let us find out what is the total height of the domain. So we have a ymax, we have a ymin that gives the difference gives me the height of the domain. Let us find out the average length of the edges. So let us say I can mark an edge like say 1 2 this is an edge or 2 3 is another edge.

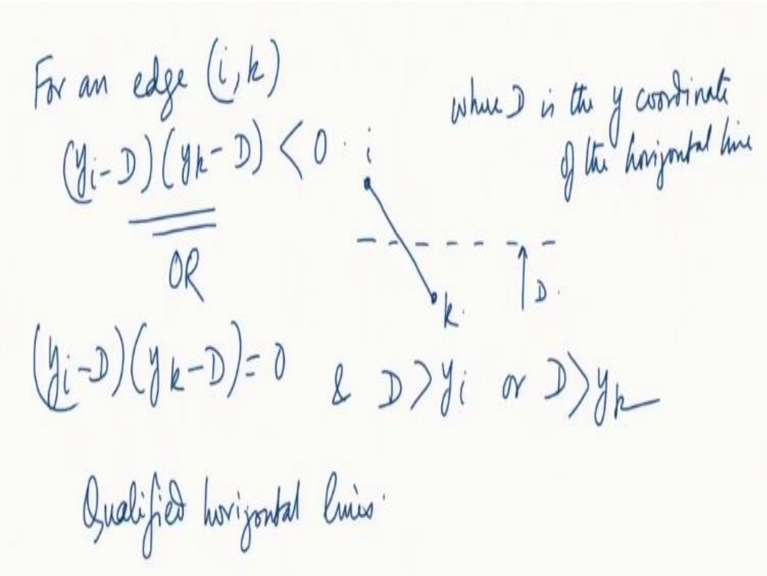
So we do a calculation by means of which we can get an average length of the edges. So if there are n number of edges, we try to do this calculation and then we would calculate the number of intervals along the y direction. Let us call that interval number as NL. So how do we go about calculating NL. We do y max minus y min by L average and we approximate it to the nearest integer say NL. What does that do?

That divides this height into an integral number of intervals, right. And of course, the final delta y that we have for each interval will be equal to y max minus y min by NL. So each one of these would have the height of delta y. Now at these respective heights, we are going to imagine horizontal lines covering the entire domain and for each horizontal line, we have to check that how many times it crosses the domain.

For example, this horizontal line crosses the domain at two locations, note that they do not need to cross the domains essentially at the nodal points, very often they would not cross the nodal points, it doesn't matter, but what matters is how many times it crosses the domain boundaries, which includes both the outer as well as internal boundaries. So this is your outer boundary, this is your internal boundary.

We are keeping an internal boundary here because in flow problems very often we have a body immersed in the flow. So the internal boundary is that immersed body and the outer boundary as though it is the outer boundary of the flow domain. So if you go to other horizontal lines, you may find that some of them are not crossing the boundary even number of times. So the question to ask is whether they are crossing even number of times? If not, then the horizontal line has to be slightly adjusted in order to make that happen.

(Refer Slide Time: 06:58)



How do you check that for an edge i, k? That means with nodes i and k there is an intersection point with any horizontal line. So what we take essentially is this. So what is D, where D is the y coordinate on the horizontal line. So it is like this that you have nodes i and k and you have a horizontal line crossing, so this coordinate is D. As long as it is satisfying this condition or it is satisfying another condition that this is exactly equal to 0 and either D is greater than yi or D is greater than yk you have an intersection point.

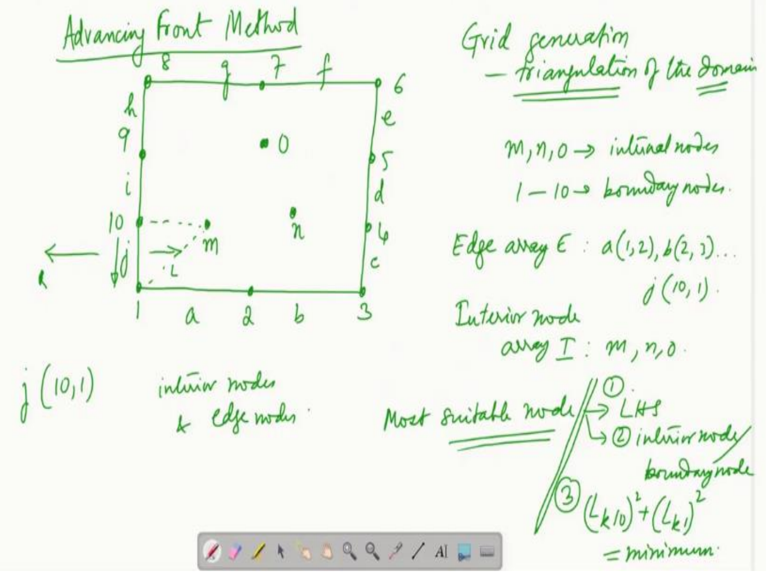
So if you are doing it through a computer program, then these logics have to be incorporated so that you can actually find out that whether these horizontal lines that you are creating are having intersection points on each one of these edges that you are searching through. This way, we create a set of qualified horizontal lines and for any qualified horizontal line, the next step that we do is we go back and if we notice that this is a qualified line.

If we call this as a qualified line it must have satisfied the criteria that we have set so there are even number of intersection points, you have 4 intersection points here. So what is the next step you do? The next step is to deploy points in between the intersection points. So you have 4 intersection points, so if you call them as A, B, C and D, the cross points, then you are incorporating new points p1, P2, P3, P4, P5.

So those points P1, P2, P3, P4, P5 are the new nodes that you have created along this horizontal line which were not existing before and remember that these are the nodes that are getting into the domain, which is bounded by the inner and outer boundary. So if that is the portion of the flow, if you imagine it that way, then you are nodalizing the flow domain. That is essentially the exercise that you are doing.

So in this manner, we have to nodalize the domain by creating a set of qualified horizontal lines and then deploying grid points in the intermediate region. So we are done with this part already.

(Refer Slide Time: 10:55)



Next thing to do is after domain nodalization, use one of the techniques for creating the cells. So we are discussing about this method, which is called advancing front method. For that I am not doing drawing the previous flow domain the way we are discussed, but I am just taking a simpler domain here for convenience of our discussion. So, let us say there are 10 boundary nodes and there is no inner body.

But there are some inner nodes which have been created through the domain nodalization that we discussed in the previous step. So m, n, o, r internal nodes and 1 to 10 are boundary nodes. Now, you have to get all this information into arrays that is very, very important. We create an edge array, let us say E. What does that contain? It contains information about each one of the edges that you have in the domain, let us call these edges like this.

So from a to j you have edges. So in edge a has nodes 1 and 2, b has 2 and 3 and then finally j has 10 and 1. So this information gets into the edge array. You have interior node array let us say I, so that contains m, n, o at the beginning of the grid generation exercise. We go to the last element in array that is we look at j which is nothing but 10, 1 and a search is conducted in order to locate nodes which are lying on the left hand side of edge j.

So if you were to stand on the edge j you would look in the direction from 10 to 1 and then you will have a right and the left to look at. So the right will point out of the domain, left will look into the domain. Note that you are orienting yourself in the direction 10 to 1. In that direction if you are orienting yourself, then this is your right and this is your left okay. So if you are looking to the left only then you will see the internal nodes.

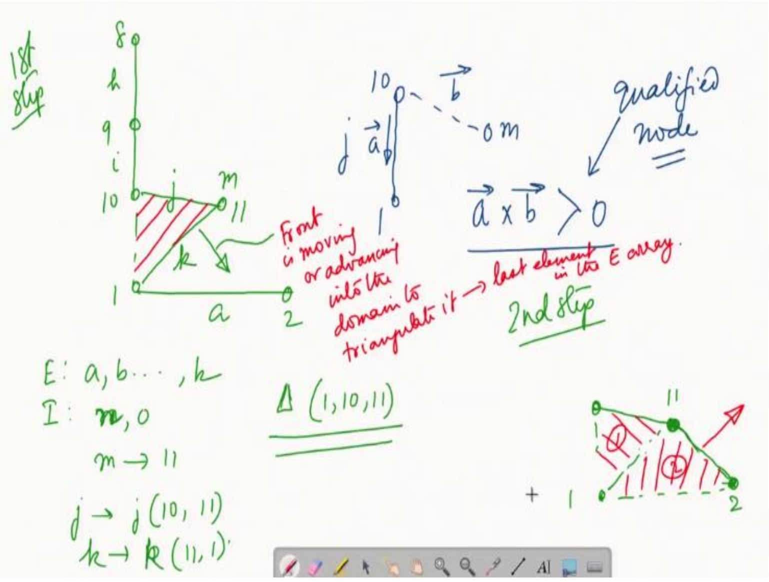
So you have to first do an exercise by means of which you search to find all nodes which are on the left hand side of this edge j. This may include both interior nodes and edge notes because as you can understand if you look to the left, you will see nodes like m, n, 4, 3, 2, 5. So it is a mix of boundary nodes and interior nodes. Now, how do you choose which is going to be the most suitable node for you as far as the triangulation of the domain is concerned?

So here when we do the grid generation, actually our objective would be to create triangles, which we would call as triangulation of the domain and that is what the unstructured grid generator is trying to do, triangulate the node, create triangular elements in the node. So the most suitable node would we defined like this that it must be lying on the left hand side that is the first criteria. It has to be an internal node that is the second criteria.

We will drop all boundary nodes because we are not trying to connect with the boundary nodes immediately, but boundary nodes can also be a part of this, not that you eliminate them altogether. More emphasis is given to another factor which is how close is that node with which you are trying to connect. So if that node is k which is the nearest node, then that kth node should satisfy this condition.

That is the distance between that node and the node 10 plus the distance from that node to the point to the node 1 squared up and summed up should be the minimum. Incidentally, in the given layout of nodes that would be satisfied by node m, which you can physically see that is the closest one and that satisfies all these conditions. So it is the most suitable mode. So what is the next step we will go ahead with?

(Refer Slide Time: 16:45)



Next step that we go ahead is something like this. We do not draw the entire diagram, but we just look at the local changes. So these are the changes which are taking place. So you have h, i, you now have a new edge j here and another k over here, k remains here as usual and as a consequence, there are a number of changes which have taken place. One is that in the edge array you have now gone up to k that has to be accommodated.

In the internal node array m is gone, you are now left with only n and o. Also the node m now becomes the eleventh node and the edges j are now defined as 10, 11, k as 11, 1. So all these changes have to be accounted in your arrays. Very important thing that has happened is a triangle comprising of these nodes have been formed. So the triangulation process has started. So you have created the first triangle in the domain and that is the objective of the whole exercise.

In between, we have missed out on an important point which we need to revisit that how did we identify the nodes which lie to our left? Coming back to that problem we just revisit the old edge j which was comprised of nodes 10 and 1, and we were checking with two vectors defined like this. We defined a vector a and the vector b. a always pointed from nodes 10 to 1, but the other vector b connected the node 10 to any one of the nodes which were being checked, whether it is a qualified node or not.

Then the idea is that once you generate these 2 vectors, you take a cross product and find out whether that cross product is more than 0. If it is more than 0, then that node becomes a qualified node because all nodes to the left would satisfy this condition based on the cross product. So that is how we search and find which are the qualified nodes. This is an important bit of information we have to keep in mind. Coming back to our triangulation process, the second step, so this was the first step in the triangulation process.

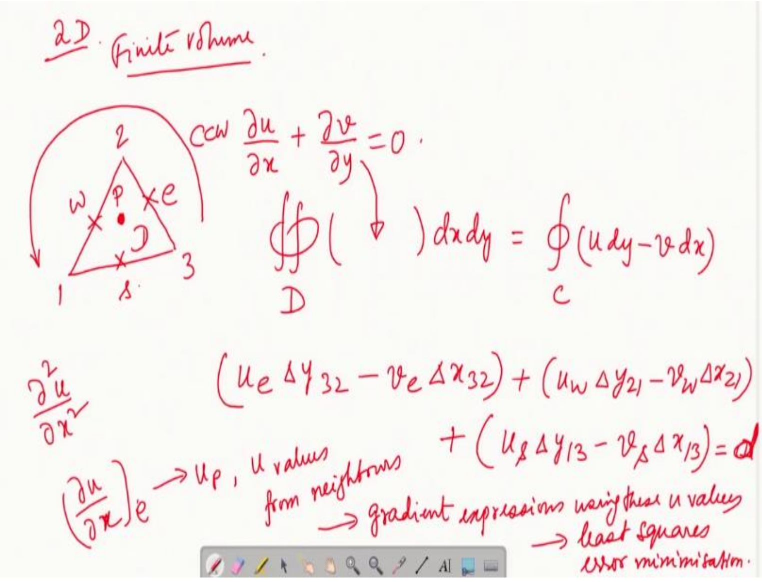
If you continue this process, the second step what could it be like? Again, we just zoom in to the local region and just see what going on where the front is moving. So we were talking about advancing front. So what is this advancing front all about? It is as though a front is moving or advancing into the domain to triangulate it and that advancing front happens to be the last element in the E array that is where the advancing front lies that is moving into the domain and triangulating it.

That is how the exercise goes on, right. So coming back to the second step let us see how it is going on. You will find that if you do the calculations, the qualified nodes, the most qualified node will turn out to be the node 2. As I said earlier that we have more emphasis on connecting internal nodes, but that does not mean we leave apart the boundary nodes. We will not be able to triangulate the domain very well if you do that.

So it is both internal as well as boundary and now comes an instance where we have chosen a boundary node and then what happens effectively is that the moment you connect that this is also going to get dotted and the new triangle that you are forming here is going to be this. So you already had formed this triangle and the second one comes up here. So this is the first triangle you formed, this is the second triangle you have formed, and the advancing front again now becomes this edge.

So this is how it is going to propagate into the domain gradually and triangulate the entire domain. So both nodalization and advancing front put together can help you develop a simple computer code to do unstructured grid generation.

(Refer Slide Time: 22:19)



And then finally if you were to solve a flow problem with that in a 2-dimensional context, you will find that finite volume will be the more appropriate approach for you on unstructured grid for obvious reasons because you do not do transformations here, you do not have the possibility of that. So if you do that, then you can look at the governing equations and try to integrate them on control volume.

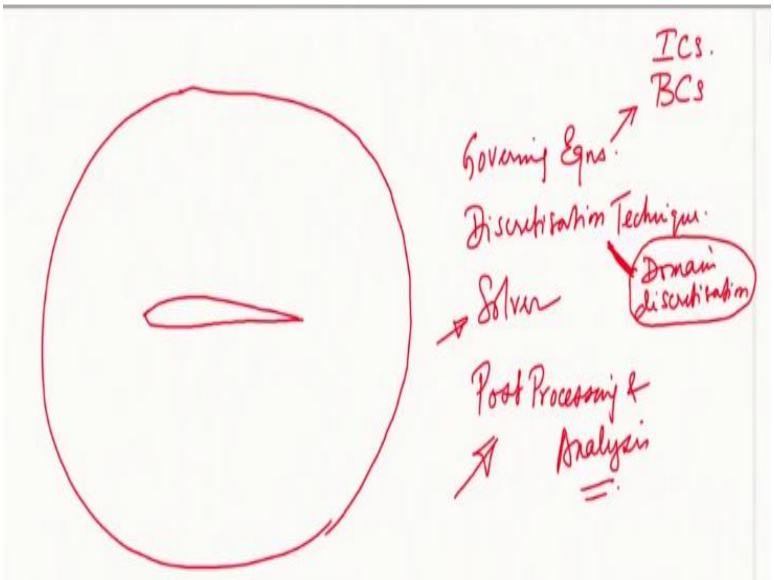
So you are creating triangles of this kind, so you can define any generic triangle like this. You can call the different edges like this and then if you take any governing equation, say the continuity equation in 2D, when you integrate it over this triangle, let us say this is the domain D. Then on integration, you write down del u del x + del v del y here and then you can show that that from surface integral it goes to a line integral and then you try to integrate it on this control volume.

So if you integrate it along this contour, you go in a counterclockwise manner and you integrate and you can write it as ue delta y 32 – ve delta x 32, that is a contribution from the east face + uw delta y 21 – vw delta x 21 + us delta y 13 – vs delta x 13. This is essentially the divergence of that cell. We should have a different nomenclature for this, so I will call it as small d let us say because we have already called the domain as capital D.

So likewise, you can take the different terms in the governing equation and convert the surface integrals into line integrals and integrate it along the periphery of such control volumes which you have created through the triangulation process. So, we have different means by which we can do the calculations in the unstructured domain and you remember that in the viscous terms, you will see second-order derivatives like this.

So when you apply the surface integral on them, they will convert to line integrals involving first-order derivatives. So you also have to find out means by which you can define first-order derivatives at these control points. So for that we generally invoke information about the u values at the cell centers from this particular cell as well as its neighbors, u values from neighbors and we try to generate gradient expressions using these u values and we generally apply least squares error minimization to do that.

(Refer Slide Time: 26:19)



Just summing up before we finish, you have now learned many of the important tools in CFD based on which you should look forward to working out problems of practical interest by using all the aspects of CFD starting from choice of governing equations, choosing a suitable discretization technique, developing a solver which handles the discretization technique, computing the flow field and performing the post processing and analysis.

So we have discussed of about these different steps in CFD and as you do that, you have to keep all the aspects in mind that what kind of partial differential equation are you handling when you are choosing your governing equations. Is it purely of elliptic, parabolic or hyperbolic type or it is a mixed type? What kind of boundary conditions and initial conditions you have to impose and the discretization technique, finite, difference, volume or element.

And also the domain discretization which comes in here, which is essentially the grid generation and a lot has to be understood about the flow itself before we do the domain discretization. So that helps us to suitably discretize keeping the grid orthogonal, keeping the grid well clustered in regions of high gradients. Then invoking a solver which works within its stability limits and dissolves the suitable length scales and timescales on the flow giving us enough details which we are desiring.

And then finally applying suitable plotting and post-processing tools so that we can extract essential and useful information from the entire exercise. So put together, all these bits and pieces are equally important and vital to make a CFD exercise successful. So wishing you all good luck with this entire exercise and I hope you enjoyed this course on CFD and it was useful for you. Thank you.