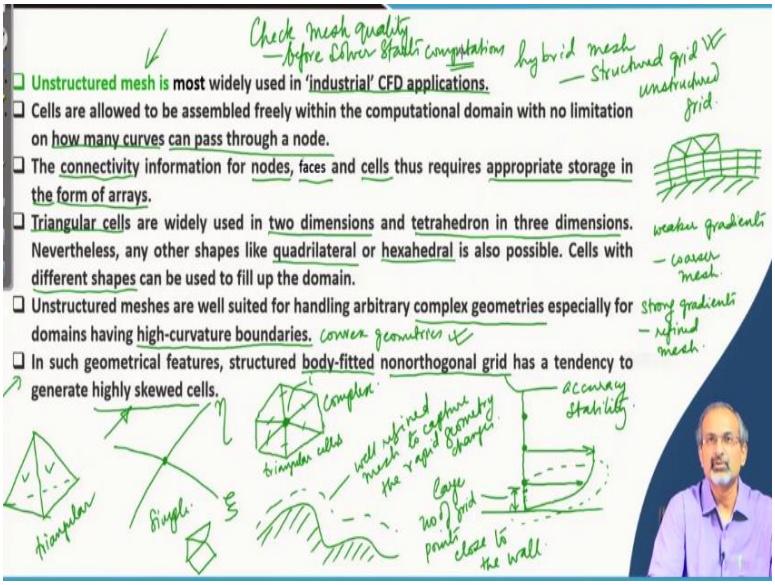
Introduction to CFD
Prof. Arnab Roy
Department of Aerospace Engineering
Indian Institute of Technology - Kharagpur

Lecture – 62
Structured and Unstructured Grid Generation - Continued

(Refer Slide Time: 00:28)



In this lecture, we continue our discussion on structured and unstructured grid generation. Last time we had discussed more on the structured mesh of the structured grid. This time also let us look at a bit of unstructured grid issues to little more detail. We take note that in the industrial CFD world, unstructured mesh happens to be the workhorse. Very often of course, they also use a so-called hybrid mesh.

Where part of the domain is discretized using a structured grid and in the remaining unstructured grid depending on complexity of geometry and also accuracy issues. We have briefly discussed about the accuracy issues in the last lecture that in general structured grids provide a higher formal order of accuracy than unstructured grid. So when you are resolving the flow close to a surface, then you might like to have a layer of structured grid close to the surface, which is well refined.

And beyond that it may be a region covered by unstructured grid and unstructured grid layers can also be coarsened quite rapidly in comparison to structured grids that can also save a lot of computational expense. So in regions where the gradients are weaker, you can afford to

have coarser mesh while strong gradients which are often produced due to rapid changes in the geometry of a body surface or viscous effects like we see in boundary layer kind of flows, you need to have a refined mesh so that you can capture the gradients well.

Just to show you an example if you have a velocity profile, which looks like this close to the wall, then you can expect that only a very refined mesh will capture the large gradients which are existing here close to the wall. So if you were to have a very, very coarse mesh where the grid points are so far apart, you will not be able to capture the gradients satisfactorily. What you need over here is far, far more refined mesh distribution, so there will be a large number of points in this region close to the wall.

Again as we said that if there are rapid geometry changes or which flow is taking place even there, you need to have very well refined mesh close to the wall to capture the rapid geometry changes. So we have to be very careful about these issues when we deploy the mesh. Going back to the issues of unstructured mesh, we see the second point says that cells are allowed to be assembled freely within the computational domain with no limitation on how many curves can pass through a node.

So this we had discussed earlier in the context of structured grid that if it is a 2-dimensional grid, then two lines from two different families can intersect at a grid point, but that kind of restriction does not exist in an unstructured mesh. So if you have a grid point or a node in an unstructured mesh, many lines can converge there. That is basically because there are many elements surrounding that node. So it must be in that portion of the grid where the grid is fairly well refined.

As a consequence, you have a lot of connectivity information which has to be built up. Connectivity information is rather simple here, but it is quite complex over here. So you have to have connectivity information with regard to nodes or grid points, faces, cells, and they have to be stored into appropriate arrays. Sometimes, we will see that in certain fluid dynamic problems, the mesh also needs to change with time.

So if you are having such a scenario, the mesh also gets enormously deformed because there may be a moving boundary and the way the boundary moves may necessitate additional of cells in some region of the flow and deletion of cells from some other region of the flow. In

that case, the connectivity information has to be rebuilt based on how the mesh is evolving. So connectivity information in terms of node is like if you call this node as the ith node, then you need to see that which are the faces that that mode chairs.

So these are the different faces that that matchers, which are the cells of which this node is apart. So this is what we mean by connectivity information. So when I call a particular node by its index, then I should have this information coming up straightaway. Only then after that, the solver can be told how to go about doing the flux calculations across all these faces which are surrounding this node and that of course involves so many different cells which are having those faces and so on.

Again, remember that not all of the domain may be filled with the same kind of geometry where we see triangular cells. You can actually have a mix and match of cells. So triangular cells of course are widely used in 2D, tetrahedral cells in 3D. So tetrahedral cells would look like this. You have a set of triangular faces. So you can see these two faces are visible and there are another two faces which are behind. So that can be kind of visualized with that dotted line.

So each one of the faces are essentially triangular. So that builds tetrahedral itself. Of course, we already discussed about using quadrilaterals or hexahedral cells in the context of structured grid generation. Even those kind of cell geometries may be used in unstructured grids and again cells with different shapes can be used to fill up the domain. For example, you may have triangular cells neighbored by quadrilaterals cells depending on geometry changes, etc. in the domain.

Complex geometries are very well tackled by unstructured measures and that is one of the major reasons why we use them in practical applications and they are also very suitable for high curvature boundaries, but one thing that we have to keep in mind is that they are very robust in terms of handling convex geometries, but when it comes to concave geometries, we have to apply them with care.

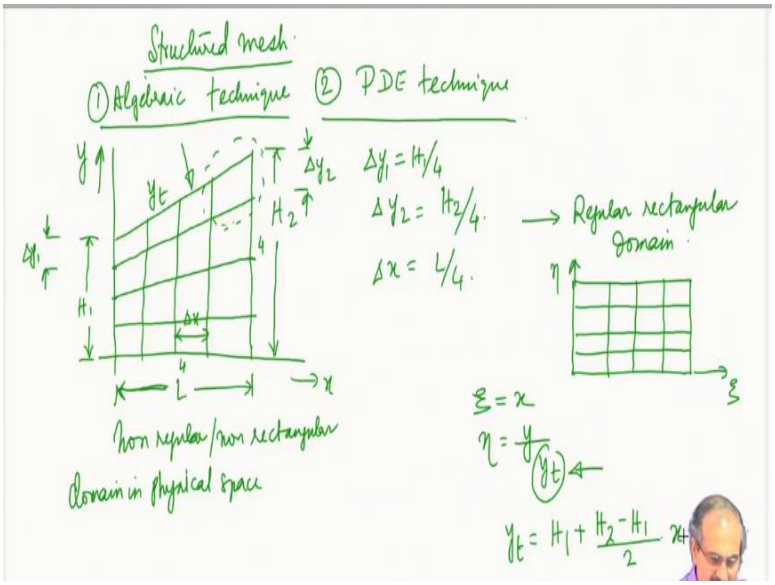
Most of the grid generators do an excellent job with convex geometries, but only the best ones handle concave geometries well. In complex geometrical features, structured body-fitted, nonorthogonal grid has a tendency to generate highly skewed cells and that is one of

the reasons why unstructured grid you know dominates over structured grid in complex geometry situations.

So first thing is that because they have to remain body-fitted, they stress themselves too much making the cells highly skewed and they also are often nonorthogonal. And we already discussed that being nonorthogonal can have adverse effects on accuracy and also stability. So there are complications which are associated with bad mesh quality. So we have to have a check on mesh quality before solver starts computations.

So if you are using some commercial or open source mesh generators be careful that there should be such steps available where you can check the mesh quality, and if the quality is not acceptable, then there are some more iterative corrections which have to be done before the mess actually comes up to an acceptable quality level before you offer it to the solver to do the computations.

(Refer Slide Time: 10:52)



Let us do some more detailing on the structured mesh which we had discussed at some length in the previous class. Today in this lecture, we are going to discuss about some techniques by which you can obtain structured measures. One is the algebraic technique, the other is partial differential equation-based technique. Let us deal with some simple examples for each one of these categories.

Let us say we have a trapezoidal domain which looks like this, and we are trying to fit a mesh in this domain. We are currently discussing it in the context of algebraic mesh generator. As

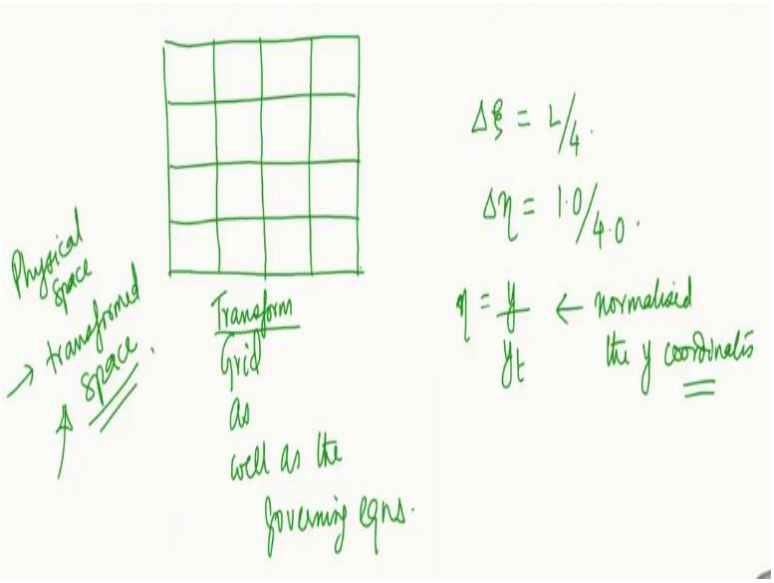
you can understand that instead of a trapezoidal domain if it was a rectangular domain for example, this would have been a trivial problem. Now, what you see is that we have fitted a 4 by 4 mesh here in this domain and we have kept uniform spacing at the boundaries.

So if you have 4 spacings here along x and 4 along the y directions on the two different edges, then these grid spacings can be indicated like this. Now, as you can understand if you watch carefully, some of these regions of the mess, there is a lot of nonorthogonality, but we are not addressing that issue right now, rather we are trying to address the issue that how we can do a simple calculation by means of which we can take this domain to a regular rectangular domain.

So, this is a non-regular or non-rectangular domain in physical space and we want it to be transformed to a regular rectangular domain in a transformed space. As we do that, what we need to understand is that when we transform, we really do not need to transform the grid along the x direction because we do not have any nonuniformity in the spacings along the x direction. The nonuniformity exists only along y.

So this is a rather simpler situation, not often seen in practical applications, but this is a good point to start. So, we would suggest a transformation like this, y top is indicated by the term yt where t stands for the top boundary and yt is essentially this upper boundary, which is defined like this.

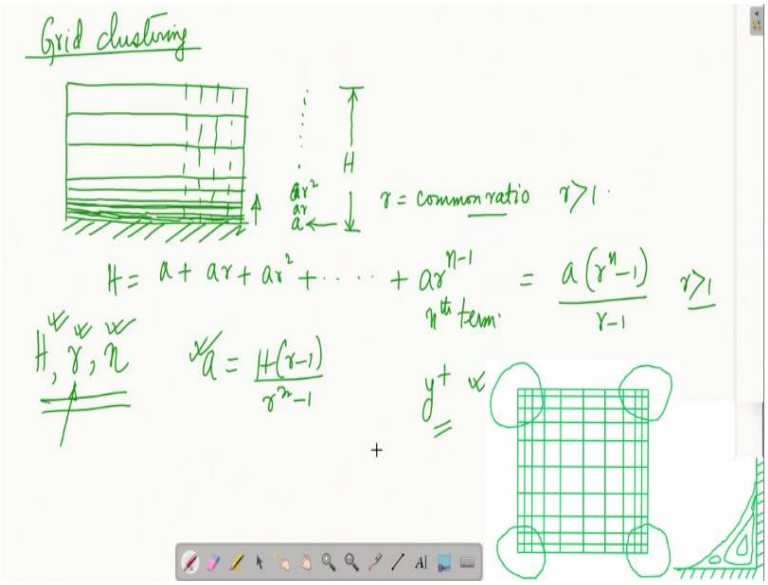
(Refer Slide Time: 16:16)



Now, if you do that transformation, what happens is you can actually generate the regular mesh like we drew in the previous diagram. That simple transformation converts the problem to a regular mesh and then in that regular mesh delta xi remains as L by 4 and delta eta remains as 1 by 4. Why is it 1 by 4? Because through the transformation, eta is equal to y by yt, you have actually normalized the y coordinates. So this is how the scaling works.

So this is the first instance which shows that you can look at the grid both in the physical space as well as a transformed space. Very often in the grid generation wall, especially in the structured grid generation wall, we tend to solve the problem by taking it to a transformed space. Again, remember that when we transform, we do not only transform the grid, we transform the grid as well as the governing equations. So transformation has to be done in coordination. So this is one of the important concepts which we will revisit shortly.

(Refer Slide Time: 18:17)



Let us take another example where we look at grid clustering. We have been talking about using grids which are well refined near boundaries in order to capture viscous effects and things like that. So, let us see how we can do it in the context of algebraic grid generation. Let us draw domain where we assume that the bottom boundary is a solid boundary and next to that we need to refine the mesh significantly.

And of course, we are looking at the mesh in a very zoomed manner so that the finest mesh levels are not well visible to us. Of course, they are all individual parallel lines, they do not cross each other, and as you can understand you can always generate the lines of the other family, but there is nothing much exciting about that. The excitement lies in this direction

that is where they are actually very well refined and then they are gradually coarsening out as you go away from the wall.

Let us assume that the thinnest layer has a thickness a, the next layer has a thickness ar, then ar square and so on, where r is a common ratio. If you have r greater than 1, then that means the thickness is gradually going to increase that way. So you have the smallest thickness multiplied by a common ratio, which is greater than 1. So, the thickness gets slightly enhanced at the next level and so on.

So if you sum up these individual cell thicknesses, you can write this as, so this is essentially the nth term and we know that this sum is given by where r is greater than 1. So let us say that we have been provided the values of H that is the height, the common ratio and the number of cells we want to put in that layer. If that is so, then you have an expression for the smallest cell thickness. Once you get that, how do you know that that is sufficient for you?

Let us say if you are doing a turbulent flow calculation. In our previous lecture on turbulence modeling, we got to learn how to calculate y+ based on the first grid point away from the wall. So that is where we actually need to check for y+. So with the a that you have obtained, you try to find out what that y+ plus is. If it is within the acceptable level, you go ahead with that choice of a. If it is not acceptable, then you try to alter some of these parameters till you get an acceptable a.

Never use too large a value of r because that creates enormous amount of stretching from one cell to the other and that gives you highly skewed cells or high aspect ratio cells, which is not good for your solver calculations that will degrade accuracy or even affect stability of the calculation in the near wall regions. So, we have to very carefully tune these parameters till we get the numbers right. Of course, this is not the only way you can cluster grids near to the surface, but this is one of the simplest ways you can try doing.

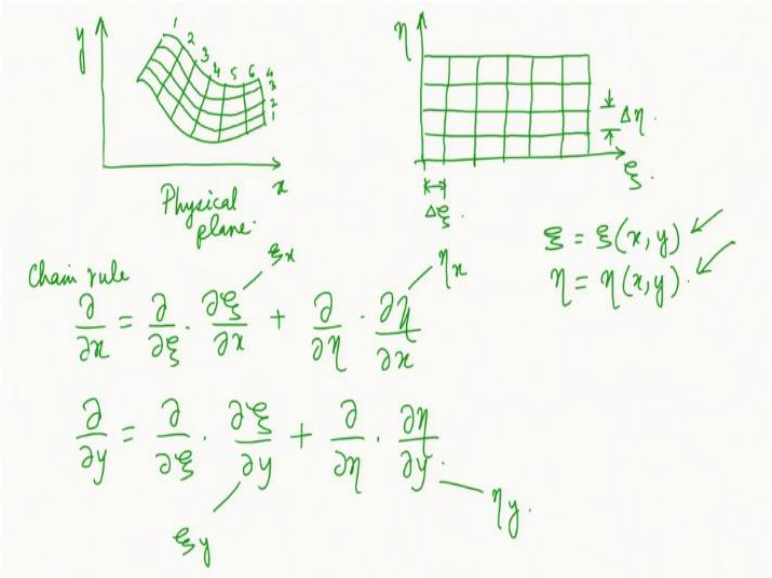
In another lecture in Navier-Stokes equations, you remember that we had discussed about a lead driven cavity problem. So you understood that because there are sharp corners in the cavity, there would be a lot of interesting flow phenomena occurring there. So that is a 2-dimensional problem where you would be interested to do the grid clustering along 2 different directions because doing it in 1 direction is not enough.

So in that case, you find certain things happening here. One is that we are applying the grid clustering both on top and bottom walls, again we are doing it on the left and the right walls. That means if you use this method, then this method has to be used repetitively as we approach each wall, so that you can get proper grid clustering in the corner regions especially. Why are we very interested to do that?

Because we know that there are going to be rich vortex structures which can be captured at these corners. If you approach those corners, you might see very small vortex structures developing like this and the larger vortex structures are beyond that and if you go to high Reynold number flows, then what happens is as we realized the other day during our turbulent modeling lectures that there is a large scale separation between the smallest eddies and the largest eddies.

So unless you have a very refined mesh, you will not be able to capture the smallest eddies. So these are very important aspects associated with the flow problem itself, which you have to realize when you do the grid generation.

(Refer Slide Time: 24:34)

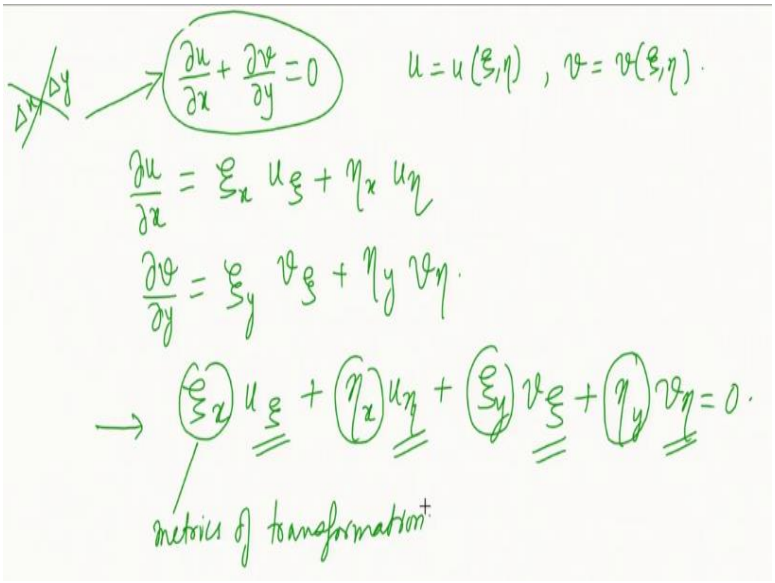


We were talking about transformations few minutes back. We need to go into the details of transformation a little further. Last time we had taken a trapezoidal domain, this time we have a more involved domain which is quite highly curved and therefore it is obvious that you cannot do a transformation along one direction and not do it along the other direction and have a nice transformation relationship buildup.

So you have to transform along both directions. So this is the domain how it looks in physical plane and now that you realize that transformation should be applied both along xi and eta, you would be interested to take it to a domain which looks like this. So we have to keep count of the number of intervals because as you can understand the transformation does not change the number of intervals along different directions.

Only thing is it kind of straightens up the geometry. So this is how it looks and you have a uniform mesh in the transform plane like this. Now that means the xi plane or rather the function xi as well as eta would be functions of x and y. So from chain rule, a partial differentiation you can write it this way that any change along x will be reflected this way in the computational domain. We can just use simpler nomenclature for these.

(Refer Slide Time: 27:28)



Once we have that, let us try applying it to a simple partial differential equation say continuity equation in 2 dimensions. Also, remember that this transformation exercise can be done for 3 dimensional problems also, only thing is it gets more involved. So if it is continuity equation, then we say that u is a function of xi, eta and so is v, and then using the chain rule expressions you can easily show, which means that the transformed continuity equation looks like this.

So realize that it will be very difficult to discretize this equation in physical space because you do not have any delta x, delta y available because it is a highly curved domain, but it gets much easier to solve it in the computational space because you are computing the velocity

derivatives with respect to xi and eta and you have a uniform mesh in the computational domain to do that. The problem remains that how you would calculate these parameters, which are often called as metrics of transformation. We will discuss more on this in the next lecture.