

**Engineering Drawing and Computer Graphics**  
**Prof. Rajaram Lakkaraju**  
**Department of Mechanical Engineering**  
**Indian Institute of Technology, Kharagpur**

**Module – 07**  
**Lecture – 52**  
**Overview of Computer Graphics – II**

Hello, everyone. Welcome to our NPTEL online certification courses on Engineering Drawing and Computer Graphics. We are covering module 17 and learning about Overview of Computer Graphics - II.

(Refer Slide Time: 00:25)

**Surface models**

CAD software packages use two basic methods for the creation of surfaces.

The first begins with construction curves (splines) from which the 3D surface is then swept (section along guide rail) or meshed (lofted) through.

The second method is direct creation of the surface with manipulation of the surface poles/control points.

A plane surface that passes through three points,  $P_1$ ,  $P_2$ , and  $P_3$  is given by  
$$P(u,v) = P_1 + u(P_2 - P_1) + v(P_3 - P_1), \quad 0 \leq u \leq 1, \quad 0 \leq v \leq 1$$

The surface normal vector when is  
$$n(u,v) = \frac{(P_2 - P_1) \times (P_3 - P_1)}{\| (P_2 - P_1) \times (P_3 - P_1) \|}, \quad 0 \leq u \leq 1, \quad 0 \leq v \leq 1$$

Once the normal unit vector is known, the surface can be also expressed in nonparametric form as  
$$(P - P_1) \cdot n = 0$$

thanks to  
Z. Jeli, B. Popokostantinovic and  
M. Stojicevic,  
Usage of 3D Computer  
Modelling in Learning  
Engineering Graphics

Planar surface    Cylindrical/conic    Sculptured

*(The slide also features a small video inset of the lecturer in the bottom right corner.)*

In the last class, we try to have some idea about these surface models and their construction especially CAD software uses two basic methods of creation of surfaces. The first one begins with the construction of curves from which the 3D surfaces then swept or meshed throughout.

The second one is the direct creation of surface with manipulation of the surface poles and control points and a computational advantage when people started using these computers or trying to use packages. The second method is most powerful because you will be having isolated discrete points and you try to construct surfaces using a different kind of curves through which.

And, one of the thing what we try to look at in the last classes as if it is a plane surface if all your data points lying on one single plane, the easiest way is trying to construct lines across these points and drawing other lines parallel to that. This is the way one can really construct a surface.

And, if we are writing it in parameter form so, the background of the or back end of your computer software actually works in that way. It picks isolated discrete points try to interpolate it and construct surfaces. And, a parametric form if you have point P 0, P 1 and P 2 any new point perhaps P one can construct in terms of your P 0 point.

P 1 differencing with P 0, and P 2 differencing with P 0. So, as a reference point P 0 we will take another point is P 1, P 2. So, we can always construct a plane surface which is passing through these three points. Four points are additional information, but three points are good enough to pass a plane and the normal to the surface is not necessary that all these three points will be on x plane or y plane, but it can be on different planes.

So, through this x, y, z different kind of locations we can pass a plane and the surface normal when you are trying to construct it will be given in terms of your P 1, P 0 points cross product with P 2, P 0 points and their norms.

So, at computer level what it does is when you are clicking a software like a pick this point, that point, another point it takes these points and your software try to do this interpolation in between those points try to put a surface. This surface can be plane surface or it can be a curved surface.

Curved surface construction is a bit more complicated and difficult compared to your plane surface and when we are going to learn about these curved surface interpolations additional concepts like Bezier curves, Coons surface and other things comes. If it is plane surface it is pretty straight forward approach.

(Refer Slide Time: 04:00)

### Surface models

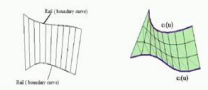
CAD software packages use two basic methods for the creation of surfaces.

The first begins with construction curves (splines) from which the 3D surface is then swept (section along guide rail) or meshed (lofted) through.

The second method is direct creation of the surface with manipulation of the surface poles/control points.

**Lofted or ruled surface**



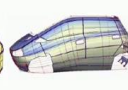
A ruled surface is generated by joining two space curves (rails) with a straight line (ruling or generator). If two curves are denoted by  $F(u)$  and  $G(u)$  respectively, for a value of  $u$ , then the parametric equation is given by

$$P(u, v) = (1-v)G(u) + vF(u), \quad 0 \leq u \leq 1, \quad 0 \leq v \leq 1$$


Planar surface

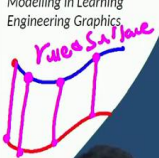
Cylindrical/conic


Sculptured

*thanks to*  
Z. Jeli, B. Popokostantinovic and M. Stojicevic,  
*Usage of 3D Computer Modelling in Learning Engineering Graphics*

*Ruled Surface*





The other way of constructing surfaces, it locally looks like plane surfaces are called ruled surfaces. For example, let us take I have some curve in 3-dimensional space similarly I have another curve. These curves might be parallel, may not be parallel; they might be offset with each other also. In that case, the easiest thing what one can do is join each and every point, perhaps pick these points to join it by a line.

Similarly, pick another point to join it by line; similarly, pick another point to join it by line and so on. If we are going to construct such kind of object that is what we called ruled surface. These curves can be arbitrary also. For example, we can have one is a straight line other one is a curve. And, we have decided to divide this entire straight line into  $n$  number of equal parts. Similarly, the other curve also we are going to divide it into  $n$  equal number of parts.

Then join each of these things, that represents a simplistic way of looking at a ruled surface and when you are doing this rule surfaces these are always straight lines the rule what we have. So, a ruled surface is generated by joining two space curves or rails is more like trains the rails what you have with a straight line ruling or generator.

If two curves are denoted by  $F$  of  $u$ , a parametric variation and  $G$  of  $u$ ; you can think of this parameter  $u$  as a function of time also. With the time the location of that point moving from one location to another location along the curve or perhaps you are defining something like it has to go along the particular surface like a curved surface, maybe it might be rolling on a sphere, it might be rolling on an ellipsoid and so on.

So, that parametric variation is  $u$  and on the parametric variation, we are defining something like a curve it goes along that the specialized direction and other curve is  $G$  of  $u$ . Then the parametric equation at the back end of that computer what it does is you construct any point in between that as some blending functions like  $1 - \nu$  multiplied by  $G$  of  $u$  plus  $\nu$  times  $F$  of  $u$  is based on how much distance you would to really look at something like the distance between  $F$  of  $u$  and  $G$  of  $u$  if we are representing a fraction in terms of  $\nu$  50 percent on this side remaining 50 percent on that side or 40 percent on this side 60 percent on that side.

That kind of weightage we will apply to really map these two curves by joining a line and try to construct a ruled surface. So, the whole objective at that back end it goes in this way. You pick lines, you try to smoothly construct a surface passing through these curves. For example, on our computer screen right now we might be having a curve of that particular shape, and perhaps there is one more line on this edge. Let us consider this is the edge.

Something like that now I want to really put the surface in that at 2-dimension level, then how does computer one of the easiest ways is? It picks these data points at the pixel level, try to interpolate how I can really join this data point that data point. So, line by line it constructs it and fills it by particular kind of colours.

The same thing happens when you are going to create an automobile or perhaps a pen or any kind of surface. For us, at the visual level, we feel like the curve or the surface has to pass in that way. But, how will you teach it to that machine the curve has to pass in that way or the tool bit when you are machining it has to go along that kind of surface?

So, there should be a geometric relation we have to teach it to these machines or to that computer and the whole objective of any surface model is the best approximation what one can get in terms of constructing a surface.

So, based on the curves what we have in 3-dimensional space we were in a position to construct a surface joining by these lines and that kind of surfaces what we call ruled surfaces.

(Refer Slide Time: 09:30)

**Surface models**

CAD software packages use two basic methods for the creation of surfaces.

The first begins with construction curves (splines) from which the 3D surface is then swept (section along guide rail) or meshed (lofted) through.

The second method is direct creation of the surface with manipulation of the surface poles/control points.

**Lofted or ruled surface**

Defining curve swept along an arbitrary spine curve

Surface Pole

Surface Spine

Linear interpolation between two edge curves

Created by lofting through cross sections

Planar surface Cylindrical/conic Sculptured

thanks to Z. Jeli, B. Popokostantinovic and M. Stojicevic, Usage of 3D Computer Modelling in Learning Engineering Graphics

NPTEI IIT Kharagpur

So, based on how many data points we would like to construct one can have a linear interpolation by creating or moving along specialized directions one of the easiest ways is dividing this line into an equal number of parts on both sides and showing one particular arrow direction and try to loft along that surfaces. If we are doing that we call that as lofted surfaces.

In the case of ruled surfaces, it is not necessary that you will have an equal number of points on both the sides, but usually, for lofter surfaces, you divide it in an equal number of points on both sides of this curve as and joined by straight lines. So, lofter surface is a specialized form of your ruled surface.

The other way of looking at this lofter surfaces, for example, we have one particular curve it is a closed curve on one side. We have another kind of closed curve also. Now, there is a line a spine curve which we call that is this one. Now, if I really move this curve along that spine curve, whatever the surface it makes that is what we call lofted surface also.

So, this curve it is a parametric curve as it is moving from this point to that point, the shape of that curve might be varying that is the reason what we why we are calling it as parametric curve maybe initially at point A it might be in circular format; at point B it might be ellipse elliptical kind of shape; at point C it might be having something like a star kind of form and so on.

So, we are moving such kind of parametric curve along a spine curve then also we will be in a position to get a particular kind of surface. That kind of surfaces what we call lofted surfaces. For example, if I am having a circle around a certain axis if I am moving maybe I might be going to get a chew.

The other way of looking at surfaces is revolved surfaces.

(Refer Slide Time: 11:55)

For example, if I have some arbitrary curve and about an axis if I am going to revolve it by 360 degrees, perhaps I might be going to get one particular shape. So, a revolved surface is generated space go about an axis of rotation. So, this is the axis, that is the curve.

When we are going to learn about this software package some of the things like defining an axis defining curve is important to construct such kind of revolved surfaces.

And, especially we have some parametric variation  $r$  of  $z$  in terms of  $u$  in terms of  $\cos$  functions  $\sin$  functions if I am going to revolve as a circle along one particular  $z$ -direction, we will get the object which we call revolved surfaces. This always is axisymmetric surface and it can be generated by rotating a plain wire form.

(Refer Slide Time: 13:16)

**Surface models**

CAD software packages use two basic methods for the creation of surfaces.

The first begins with construction curves (splines) from which the 3D surface is then swept (section along guide rail) or meshed (lofted) through.

The second method is direct creation of the surface with manipulation of the surface poles/control points.

✓ Tabulated cylinder surface

Project curve along a vector  
in SolidWorks, created by extrusion

Extrude

Defining curve swept along an arbitrary spine curve  
Swept surface

Planar surface   Cylindrical/conic   Sculptured

*thanks to*  
Z. Jeli, B. Popokostantinovic and M. Stojicevic,  
Usage of 3D Computer Modelling in Learning Engineering Graphics

*[Video inset of a presenter]*

Usually, we define positive rotation, something like in counterclockwise direction we will try to rotate the objects. Other surfaces are called tabulated cylinder surfaces; in that case, we have one particular curve. If I am dragging that along the particular direction it forms a surface, that kind of things what we call this tabulated surfaces and especially, a software like SolidWorks which we will learn it in next classes, there is a command named extrude or extrusion. Using that command once you draw a curve you can really sweep it in the lateral direction to make the object.

(Refer Slide Time: 14:16)

**Interpolation methods for Surface construction**

- Hermite bicubic surfaces
- Bezier surfaces
- B-spline surfaces
- Coons surfaces
- Gordon surfaces
- Fillet surfaces
- Offset surfaces

In many applications such as computer visions, medical imaging, and image generation, surface data is available in the form of set of points, planar or 3-dimensional contours and it is desired to reconstruct a surface from this data

*thanks to*  
<http://graphics.cs.cmu.edu>

NPTEL IIT Kharagpur

Now, where do we use such kind of surfaces? Are there any better way of representing these surfaces? In many applications, if we are looking at like computer visions like about constructing these 3-dimensional models and so on, or medical imaging.

Perhaps you might be having a medical imaging thing and you would like to construct 3-dimensional pictures from different views, is not possible by a human being to visualize that complicated medical images to construct something like 3D.

So, we have to teach it to the computer or perhaps use at least a software which can read this different views try to map that construct the surface and the back end what happens is you impose certain mathematical functions which pass through this data points because we have isolated data points in terms of pixels like colours red, blue, green or perhaps white and black, grey grayscale kind of images and so on. So, we have colour contrast from there we will try to impose certain functions pass curves surfaces through that to create something like an object for example, like MRI scan how the software constructs that 3D images. So, for medical imaging image generation surface data is available only in the form of a set of data points and what we all the time interested is constructing that 3D image.

And, especially most of these software work on basic principles like maybe going with Hermite bicubic surface construction processors or Bezier's surfaces or B-spline surfaces something like, Coons surfaces which are popular or Gordon surfaces - sudden sharp changes associated with surfaces, something like rounding of surfaces like fillet surfaces, something like chopping off these materials named offset surfaces. So, now we will learn we will have some idea about these surface construction procedure step by step.

(Refer Slide Time: 16:47)

**Interpolation methods for Surface construction**

- Hermite bicubic surfaces
- Bezier surfaces
- B-spline surfaces
- Coons surfaces
- Gordon surfaces
- Fillet surfaces
- Offset surfaces

**Curves**

- Curves: single parameter  $u$  (e.g. time)
- $x = x(u), y = y(u), z = z(u)$
- Circle:  $x = \cos(u), y = \sin(u), z = 0$
- Tangent described by derivative

$$p(u) = \begin{bmatrix} x(u) \\ y(u) \\ z(u) \end{bmatrix} \quad \frac{dp(u)}{du} = \begin{bmatrix} \frac{dx(u)}{du} \\ \frac{dy(u)}{du} \\ \frac{dz(u)}{du} \end{bmatrix}$$

**surfaces**

- Use parameters  $u$  and  $v$
- $x = x(u,v), y = y(u,v), z = z(u,v)$
- Describes surface as both  $u$  and  $v$  vary
- Partial derivatives describe tangent plane at each point  $p(u,v) = [x(u,v), y(u,v), z(u,v)]$

$$\frac{\partial p(u,v)}{\partial u} = \begin{bmatrix} \frac{\partial x(u,v)}{\partial u} \\ \frac{\partial y(u,v)}{\partial u} \\ \frac{\partial z(u,v)}{\partial u} \end{bmatrix} \quad \frac{\partial p(u,v)}{\partial v} = \begin{bmatrix} \frac{\partial x(u,v)}{\partial v} \\ \frac{\partial y(u,v)}{\partial v} \\ \frac{\partial z(u,v)}{\partial v} \end{bmatrix}$$

thanks to <http://graphics.cs.cmu.edu>

First of all, we have to define something named curves. Curves are always single parameter representation. You may be having  $x$  as a function of  $x$  of  $u$ ;  $y$  might be a parametric variation  $y$  as a function of  $u$ ;  $z$  might be a function of  $u$ . If it is something like a spiral you have  $r$  theta phi kind of coordinates which can be connected by one single parameter.

In the same way, you have any point  $x, y, z$  where you have information maybe one can pass a curve through that and that parametric variation what we are saying referring to  $u$ . For example, if it is a circle  $\cos u \sin u$  defines your circle. This  $u$  can be theta angular coordinate in one particular instance case and because it is a circle 2-dimensional thing. So, there will not be any  $z$  variation. And, now, I would like to construct something like a curve I had to join these data points; that means, I need something like from one point to other points if I am moving how that tangent looks like. First of all, this is the first data point. Unless I know the tangent information I do not know where exactly the third, fourth, fifth, sixth, seventh points are present.

So, all the time I need information about the tangent, the point information the tangent information so that I can sweep in the next direction to represent something like a curve which can be fit in a smooth way as a circle. So, we require that  $P$  information and also we require the tangent information, if we have we will be in a position to draw a smooth curve.

It is more like on drawing sheets earlier we have to try to draw smooth freehand curves. That is because our visual says that from this point you go in that direction, but to the computer, you have to teach construct a tangent, go to next point by integration maybe it might be an Euler integration, go join those things and construct a curve.



If it is a surface we will be having two parameters  $u$  and  $v$ ; maybe in two directions we will have this parametric variation  $u, v$  and any data point on that surface represented by two parameters  $u, v$ ;  $y$  is also as a function of  $u, v$ ;  $z$  also as a function of  $u, v$ . And, we require partial derivatives describing tangent at those points like how we talked about a curve point and a tangent so that I can construct a circle.

If I would like to construct a surface first point, the second point, perhaps third point fourth point these are the points we have. One way is you can always connect by lines, but you want better description naturally you require additional information on the curve has to go the downward direction in that way, the curve has to go upward direction, the curve has to go upward direction and downward direction.

Then you will be in a position to join that by in that way. So, this kind of principle you have to teach it to computer; that means, you require the partial derivatives of that surface which we are describing  $P$  of  $u$  comma  $v$ .

(Refer Slide Time: 20:51)

**Interpolation methods for Surface construction**

- Hermite bicubic surfaces
- Bezier surfaces
- B-spline surfaces
- Coons surfaces
- Gordon surfaces
- Fillet surfaces
- Offset surfaces

**Curves**

- Curves: single parameter  $u$  (e.g. time)
- $x = x(u), y = y(u), z = z(u)$
- Circle:  $x = \cos(u), y = \sin(u), z = 0$
- Tangent described by derivative

$$p(u) = \begin{bmatrix} x(u) \\ y(u) \\ z(u) \end{bmatrix} \quad \frac{dp(u)}{du} = \begin{bmatrix} \frac{dx(u)}{du} \\ \frac{dy(u)}{du} \\ \frac{dz(u)}{du} \end{bmatrix}$$

**surfaces**

- Use parameters  $u$  and  $v$
- $x = x(u,v), y = y(u,v), z = z(u,v)$
- Describes surface as both  $u$  and  $v$  vary
- Partial derivatives describe tangent plane at each point  $p(u,v) = [x(u,v), y(u,v), z(u,v)]$

$$\frac{\partial p(u,v)}{\partial u} = \begin{bmatrix} \frac{\partial x(u,v)}{\partial u} \\ \frac{\partial y(u,v)}{\partial u} \\ \frac{\partial z(u,v)}{\partial u} \end{bmatrix} \quad \frac{\partial p(u,v)}{\partial v} = \begin{bmatrix} \frac{\partial x(u,v)}{\partial v} \\ \frac{\partial y(u,v)}{\partial v} \\ \frac{\partial z(u,v)}{\partial v} \end{bmatrix}$$

thanks to <http://graphics.cs.cmu.edu>

$u, v$   
 $P$   
 $\frac{\partial P}{\partial u}$   
 $\frac{\partial P}{\partial v}$

NPTEL IIT Kharagpur

You can think of this  $u, v$  as the new coordinate system on which you are going to draw a surface  $P$  where  $\text{del } P$  by  $\text{del } u$  and  $\text{del } P$  by  $\text{del } v$  you need to know or you have to evaluate at that points.

So, you might be having 4 data points, but these tangents you are going to map or construct extra interpolate, extra 30 more points in between these 4 points. What is the best way of representing that, that is the whole objective of this surface construction.

(Refer Slide Time: 21:28)

**Interpolation methods for Surface construction**

- Hermite bicubic surfaces
- Bezier surfaces
- B-spline surfaces
- Coons surfaces
- Gordon surfaces
- Fillet surfaces
- Offset surfaces

**Curves**

- Curves: single parameter  $u$  (e.g. time)
- $x = x(u), y = y(u), z = z(u)$
- Circle:  $x = \cos(u), y = \sin(u), z = 0$
- Tangent described by derivative

$$p(u) = \begin{bmatrix} x(u) \\ y(u) \\ z(u) \end{bmatrix} \quad \frac{dp(u)}{du} = \begin{bmatrix} \frac{dx(u)}{du} \\ \frac{dy(u)}{du} \\ \frac{dz(u)}{du} \end{bmatrix}$$

Parameters often have natural meaning

- Easy to define and calculate
- Tangent and normal
- Curves segments (for example,  $0 \leq u \leq 1$ )
- Surface patches (for example,  $0 \leq u, v \leq 1$ )

**surfaces**

- Use parameters  $u$  and  $v$
- $x = x(u, v), y = y(u, v), z = z(u, v)$
- Describes surface as both  $u$  and  $v$  vary
- Partial derivatives describe tangent plane at each point  $p(u, v) = [x(u, v) \ y(u, v) \ z(u, v)]^T$

$$\frac{\partial p(u, v)}{\partial u} = \begin{bmatrix} \frac{\partial x(u, v)}{\partial u} \\ \frac{\partial y(u, v)}{\partial u} \\ \frac{\partial z(u, v)}{\partial u} \end{bmatrix} \quad \frac{\partial p(u, v)}{\partial v} = \begin{bmatrix} \frac{\partial x(u, v)}{\partial v} \\ \frac{\partial y(u, v)}{\partial v} \\ \frac{\partial z(u, v)}{\partial v} \end{bmatrix}$$

**Curve fitting**

- Restrict  $x(u), y(u), z(u)$  to be polynomial in  $u$
- Fix degree  $n$
- Each  $c_k$  is a column vector

$$p(u) = \sum_{k=0}^n c_k u^k$$

$$c_k = \begin{bmatrix} c_{1k} \\ c_{2k} \\ c_{3k} \end{bmatrix}$$

*thanks to*  
<http://graphics.cs.cmu.edu>

*C<sub>1</sub>u<sup>1</sup> + C<sub>2</sub>u<sup>2</sup> + C<sub>3</sub>u<sup>3</sup> + ... + C<sub>n</sub>u<sup>n</sup>*

Some of the essential information what one should know is about tangent and normals, and what is the range these  $u$  and  $v$  information one will be having. For example, your space might be from 1 meter to extend it to something like 500 meters. But, when you are teaching it to the computer the modules when you are going to develop or perhaps the background program you always normalize this one 500 to something like 0 1 kind of coordinate system.

You construct any data point it is more like a mapping from 1 to 500 to 0 to 1 kind of system. When you do that you are shrinking the same submodule or function you are all the time calling these are my  $x$  limits,  $y$  limits. So, fit a curve in between that or fit a surface in between 0, 1 and again rescale it up to 1 to 500. That is the way it works.

If it is a curve-fitting either we will use the algebraic relations in terms of linear relations or perhaps we will be using polynomial functions. For example, a curve  $x$  of  $u$ ,  $y$  of  $u$ ,  $z$  of  $u$  the point we have parametric variation, if I am going to fix a polynomial in  $u$  then the curve  $P$  is the curve in parametric form can be written as  $c_k$  multiplied by  $u^k$ , it is more like the summation  $k$  is equal to 0 to  $n$  means  $c_1 u^1$  plus  $c_2 u^2$  plus  $c_3 u^3$  plus and so on how many data points you have that much.

Based on your tangents you try to derive a relation what might be the weight one has to use for  $c_1, c_2, c_3$  and so on; whether it should be 0.5 times of  $u^1$ , whether 0.1 times of that or other things. The best example is when we are decomposing a function in terms of sinusoidal cosine kind of thing by using Fourier series, we always write  $a_0, a_1, a_2, a_3$  and so on;  $b_0, b_1, b_2, b_3$  and so on so things.

And, we try to impose certain kind of condition what should be that factor a 0, a 1, a 2 and so on b 0, b 1, b 2 based on the discrete points what we have information usually this Fourier series like sine cosine we use it for periodic kind of functions, but these data points may not follow periodic functions. In that case, the best way of approximating this curve is by polynomial expansions.

And, we minimize maximize certain kind of weights so that we carefully adjust that everything does by computer by programs and finally, it constructs a smooth curve along with these data points.

(Refer Slide Time: 24:57)

**Interpolation methods for Surface construction**

- Hermite bicubic surfaces
- Bezier surfaces
- B-spline surfaces
- Coons surfaces
- Gordon surfaces
- Fillet surfaces
- Offset surfaces

**Curves**

- Curves: single parameter  $u$  (e.g. time)
- $x = x(u), y = y(u), z = z(u)$
- Circle:  $x = \cos(u), y = \sin(u), z = 0$
- Tangent described by derivative

$$p(u) = \begin{bmatrix} x(u) \\ y(u) \\ z(u) \end{bmatrix} \quad \frac{dp(u)}{du} = \begin{bmatrix} \frac{dx(u)}{du} \\ \frac{dy(u)}{du} \\ \frac{dz(u)}{du} \end{bmatrix}$$

**surfaces**

- Use parameters  $u$  and  $v$
- $x = x(u,v), y = y(u,v), z = z(u,v)$
- Describes surface as both  $u$  and  $v$  vary
- Partial derivatives describe tangent plane at each point  $p(u,v) = [x(u,v) \ y(u,v) \ z(u,v)]^T$

$$\frac{\partial p(u,v)}{\partial u} = \begin{bmatrix} \frac{\partial x(u,v)}{\partial u} \\ \frac{\partial y(u,v)}{\partial u} \\ \frac{\partial z(u,v)}{\partial u} \end{bmatrix} \quad \frac{\partial p(u,v)}{\partial v} = \begin{bmatrix} \frac{\partial x(u,v)}{\partial v} \\ \frac{\partial y(u,v)}{\partial v} \\ \frac{\partial z(u,v)}{\partial v} \end{bmatrix}$$

**Surface fitting**

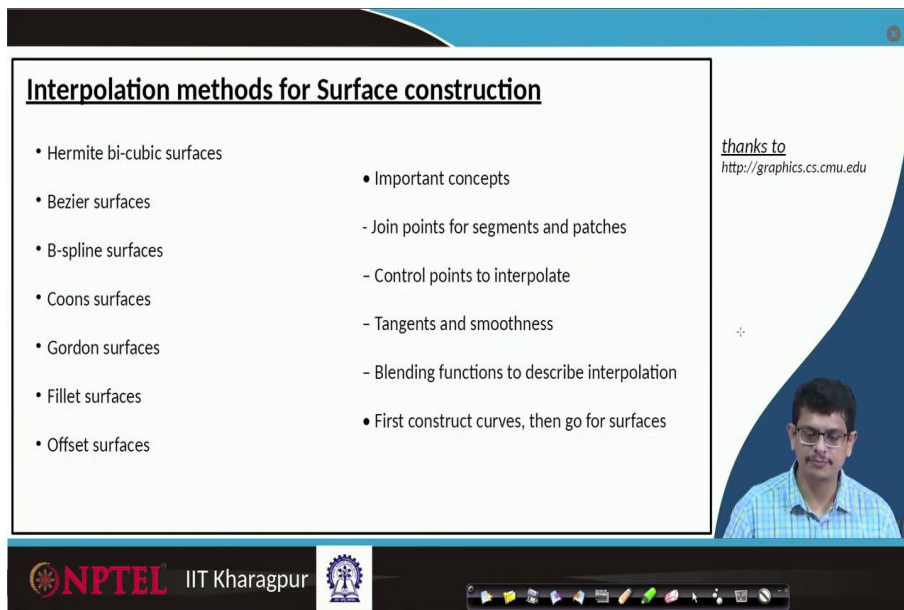
- Restrict  $x(u,v), y(u,v), z(u,v)$  to be polynomial of fixed degree  $n$
- $p(u,v) = \begin{bmatrix} x(u,v) \\ y(u,v) \\ z(u,v) \end{bmatrix} = \sum_{i=0}^n \sum_{k=0}^n c_{i,k} u^i v^k$
- Each  $c_{i,k}$  is a 3-element column vector
- Restrict to simple case where  $0 \leq u,v \leq 1$

*Handwritten notes:*  
 thanks to <http://graphics.cs.cmu.edu>  
 $c(i,k) \ u(i) \ v(k)$

Similarly, if we are going to construct surfaces this polynomial expansion we will not only just do in one dimension like  $x$ , but we might be going to do it in  $x, y$  directions our  $u, v$  parametric variations. So, if we are looking at that carefully the polynomial surface  $P$  as a function of  $u, v$  represented in terms of your  $x, y, z$  points throughout this space whatever those discrete points we have can be expanded in terms of double summation.

Summation on  $i$ , summation on  $k$  it is more like if you are writing a program for  $i$  is equal to 1 to or for  $i$  is equal to 0 to  $m$  minus 1 data points; for  $j$  is equal to 0 to  $n$  minus 1 data points, evaluate some function  $c$  of  $i$  comma  $k$  multiplied by your parametric  $u$  of  $i$  multiplied by  $v$  of  $k$  that is the way we expand that and try to fix  $c_{i,k}$  informations. And, based on  $i, k$  kind of information how many data points I would like to have, I will be in a position to construct a surface.

(Refer Slide Time: 26:15)



The slide is titled "Interpolation methods for Surface construction" and is presented in a video lecture format. It features a list of interpolation methods on the left and a list of important concepts on the right. A small inset video of the presenter is visible in the bottom right corner of the slide area. The slide also includes a thank you note and a URL.

**Interpolation methods for Surface construction**

- Hermite bi-cubic surfaces
- Bezier surfaces
- B-spline surfaces
- Coons surfaces
- Gordon surfaces
- Fillet surfaces
- Offset surfaces

**Important concepts**

- Join points for segments and patches
- Control points to interpolate
- Tangents and smoothness
- Blending functions to describe interpolation

• First construct curves, then go for surfaces

*thanks to*  
<http://graphics.cs.cmu.edu>

NPTEL IIT Kharagpur

So, some of the important concepts for this surface constructions are joined points for segments and patches either you join by one point to other point or by patches we will do. The minimum data points what we have those we will use it like control parameters to make it up and down kind of things and we require something about tangents, their smoothness also.

Based on that information we will blend the functions to describe the interpolations and first, we will always construct curves and then we will go with surfaces. So, any software what we use it always involves such kind of step by step interpolations.

So, in the next class, we will in detail learn about these different kinds of surfaces like Hermite, Bezier and other surfaces.

Thank you very much.