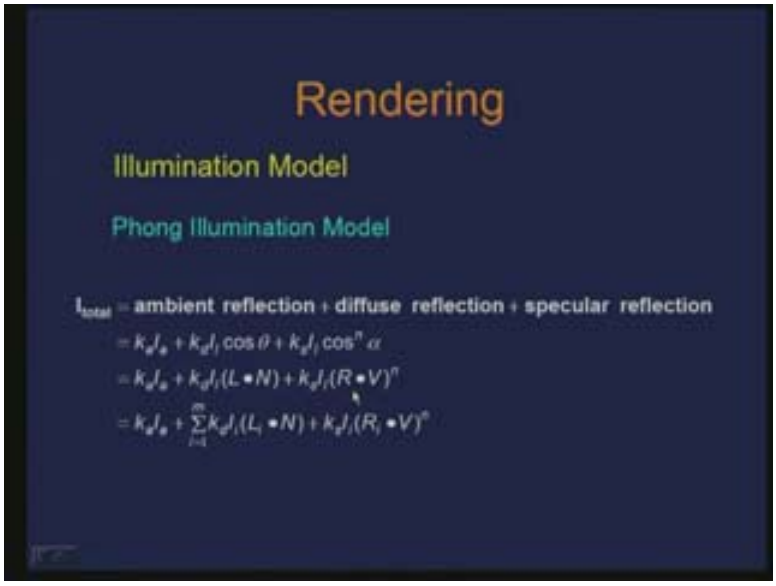
Introduction to Computer Graphics
Dr. Prem Kalra
Department of Computer Science and Engineering
Indian Institute of Technology, Delhi
Lecture - 22
Rendering (Contd….)

In continuation of rendering of surfaces today let us look at how we shade polygonal surfaces. Last time we basically looked at the illumination model where the intensity which is computed at a point on a surface has primarily three components to be computed. One is the ambient reflection, the diffuse reflection and the specular reflection. And each of these components are basically computed, for instance diffuse reflection is computed using Lamberts law where the reflected intensity is proportional to the angle of the normal vector and the light vector. Then we also have a diffuse coefficient which is Kd which also captures the material property of the surface.

Similarly, specular component in fact captures the shininess or the highlights on the surface. So, for that as opposed to lamberts surface or surfaces which obeys Lamberts law where the intensity is only proportional to the angle between the light vector and normal vector. Therefore it is independent of the viewing direction and the specular component is actually dependent on the viewing direction. So, that is where the role of the viewing vector comes and there is a reflected vector of the light.

(Refer Slide Time: 00:04:46)



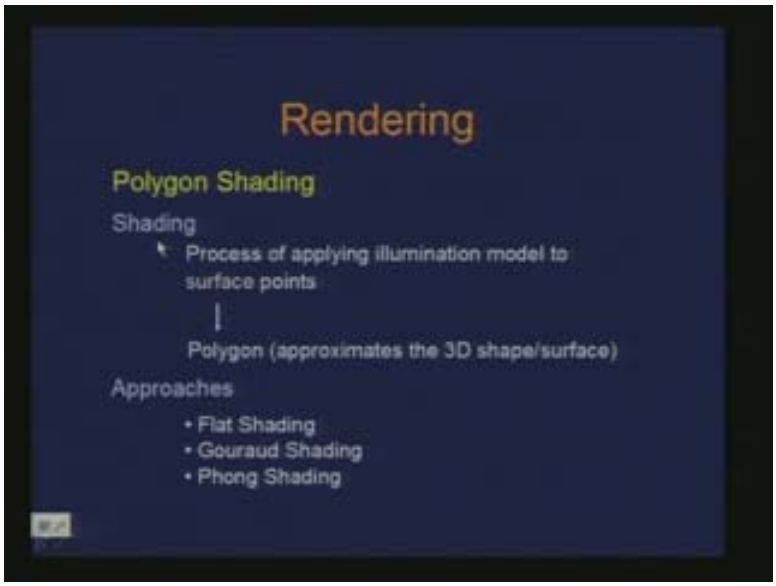
So using these two vectors we can compute the specular component of the illumination. And in fact we also looked at how we compute reflected vector in terms of L and N. and there is also this specular coefficient which again captures in some sense the material property on which this specular component is computed.

The exponent n is an indicator of the concentration of the intensity or the shininess which we have computed. And the ambient illumination may have the incident intensity Ia which could be taken as different form the light source which has been included for the illumination. And this says that we can handle multiple light sources which is just a matter of considering each light source and computing the component for diffuse and specular. This basically tells you how to compute the illumination at a point.

Most of the time the surfaces are modeled using a collection of polygons for various reasons but one of the reasons is that the shading calculation or the illumination calculation used for polygons is hardware supported. Therefore you have a faster way to computer the shade.

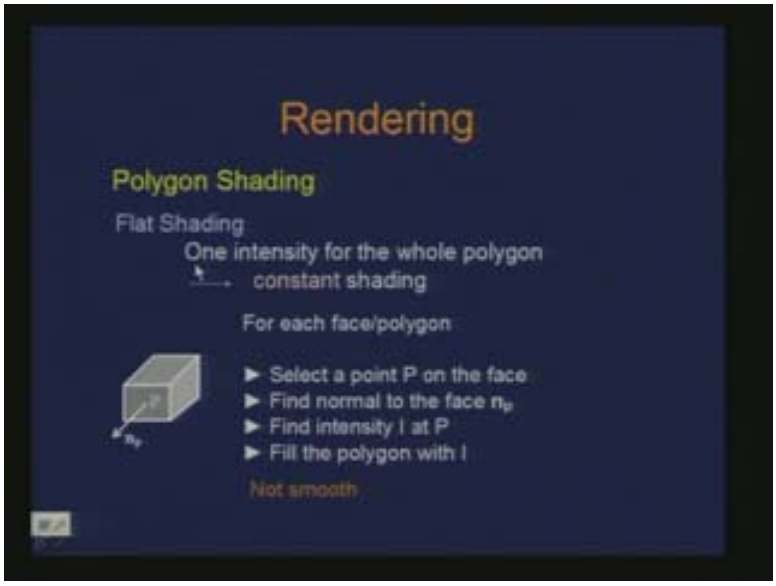
Today we are going to look at the models which permit us to do shading for polygonal surfaces. Firstly we learn on how to shade polygons. Shading is nothing but a process of applying the illumination model to surface points. In general this is what shading is. And when we model the surface as a polygon which is nothing but some sort of an approximation to the 3D shape of the surface then we are basically referring to polygon shading.

(Refer Slide Time: 00:06:15)



There are basically three approaches which are used for shading polygons. One is flat shading, then gouraud shading and phong shading. For flat shading we actually have one illumination or intensity computation for the entire polygon. The polygon is assigned one intensity. That is where we have constant shading for the polygon. That is why sometimes flat shading is also called as constant shading.

(Refer Slide Time: 7:45)



What is involved in this constant shading or flat shading? For each face of the polygon we select a point p on the polygon or on the face, in many cases we consider this point p to be the centroid of the points of the vertices of the polygon. So this is one point on the face. Then we find out or compute normal to the face. And then given the normal now we can do computation of the illumination or the intensity at that point p and then we use that intensity for filling the entire polygon. So, typically if you have an object of this kind then this whole face will have one intensity, this whole face will have one intensity and this face will have one intensity. So, computationally it is quite simple not much is involved but we observe that the surface which gets rendered is not smooth.

We see the discontinuity in intensity then we go from one face to another face which may be a desirable feature is certain objects. For instance, particularly when we are looking at an object of the kind cube then having an intensity for each face as something what we wanted to see but there are objects which are more like an approximation of the continuous surface which is decomposed into collection of polygon or triangles there we would like a smooth change of intensity.

One variation of this flat shading is that instead of considering any point p on the face which I said could possibly be the centered of the face one could consider every point on the face. so there we are trying to count for the variation in the distance which we are going to the compute for the light vector or the light vector itself because that is the vector from the point on the surface to the light source so that variation is possible. Sometimes that is also called as facet shading but the normal one is the same.

Here is an example: This is again the famous [.....] tea pot made up of Beizer patches. Here the tea pot surface is basically broken down into several polygons. Here this is a polygon, this is polygon, this is a polygon so the polygonal structure is visible and for

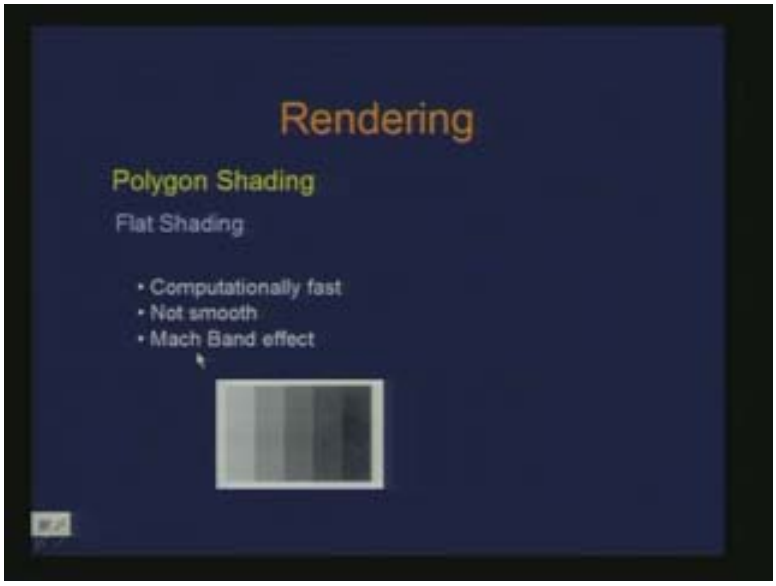
each of the polygons we employ flat shading and that is where we see that the edges are quite sharp.

(Refer Slide Time: 00:11:37)

So as a rendering or the display of the object in terms of its realism is not as realistic as we would like to have it particularly when we know that this was derived from the continuous surface representation, parametric representation. One option could be to have very small polygons then try to use flat shading for each of those small polygons. Clearly there is an overhead but then you have to treat so many polygons.

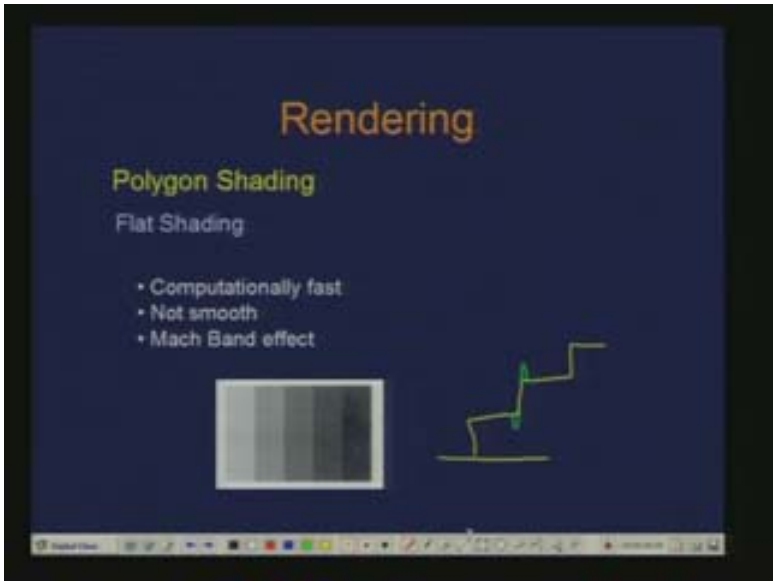
Let us see other techniques which would give you better results so that you need not have too many polygons. Computationally it is very simple and so it is fast. Just a single calculation with respect to the normal we have for the face and we also observe that it is not smooth. There is something called Mack band effect. It is actually pertaining to the perceived intensity by our visual system. Particularly it refers to where we have discontinuity at the boundaries. So, if I have this band of intensities, it is lighter to darker side. So when you look at the boundaries here you have the perception that it is much brighter on this side and it is darker on the other side. So we perceive what happens at this boundary in a slightly different manner.

(Refer Slide Time: 00:14:03)



Hence I try to plot the step function we have for these intensities. So what happens is the perception at these boundaries for instance here is actually like this and here it is like this.

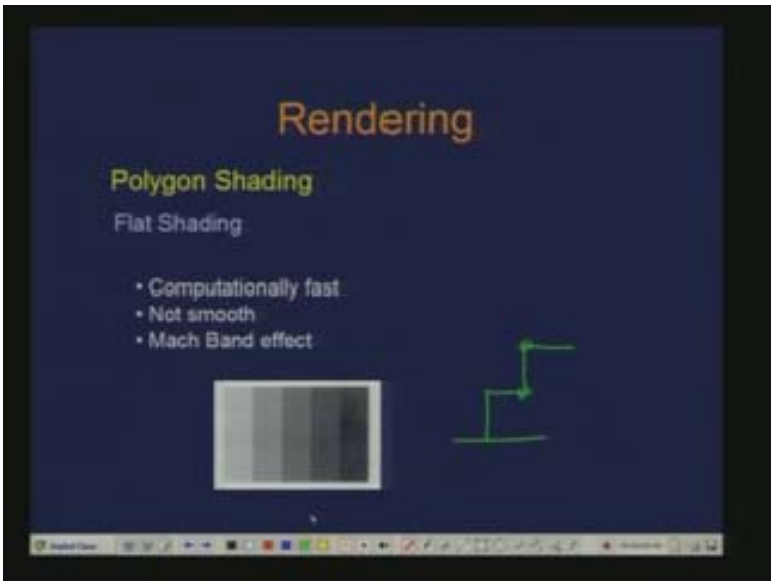
(Refer Slide Time: 00:15:47)



What you see here looks like this particular point had lower intensity than this band and this point of point in the vicinity this border has higher intensity. Therefore these boundaries are perceived to have this artificial brighter spots and darker spots. This is due to the perceived intensity from our visual system. When we are talking about the flat shading flat shading is shading where we have discontinuity at the boundaries. Therefore

the Mack band effect in some sense pronounced. So if you try to have smoother boundaries this is going to be reduced.

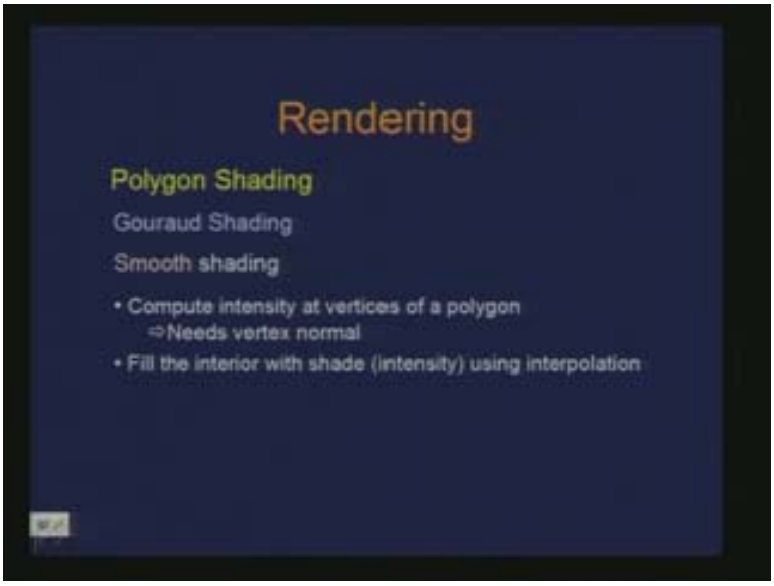
(Refer Slide Time: 00:18:14)



This is the change we will have so here and here these are the places the perceived intensity is different from these, this is the actual intensity. The cos of the difference between this and this just in the vicinity of this I see a [dip] here and this is a visual perception. For instance, here is a slight dip on the lighter side so that the boundaries have a different perception of intensity so here on this side it is slightly darker and on this side it is slightly lighter that it belongs to. That is completely a visual effect, it is a perception but is not the actual thing. But if we are using flat shade then this effect is going to be pronounced.

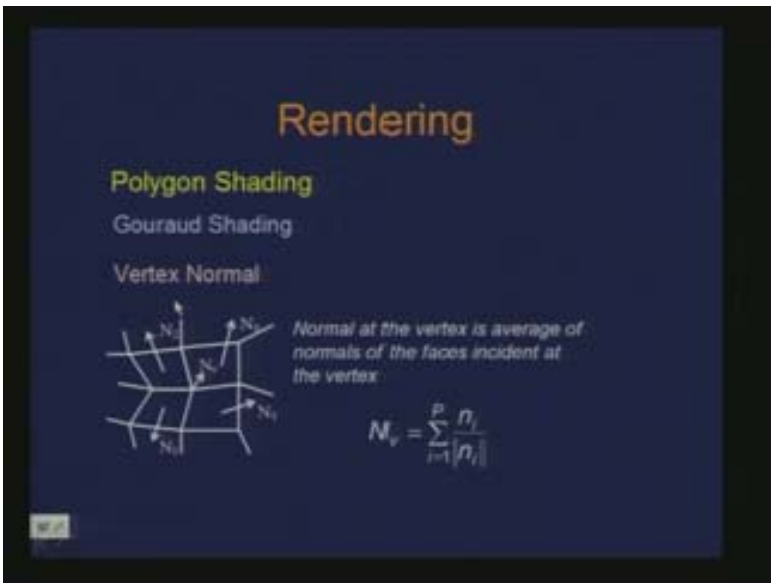
Shading methods:
Here we compute the intensity at vertices of polygon. For flat shading we computed intensity for the polygon as face. Here we compute the intensity at the vertices of the polygon so this requires computation of the vertex.

(Refer Slide Time: 00:19:59)



Now instead of having face normal I need a vertex because I am going to compute the intensity at the vertices of the polygon. And once I have done this, that means I have computed the intensity at the vertices of the polygon I can then fill the interior with the shade of the intensity using some interpolation.

(Refer Slide Time: 00:22:08)



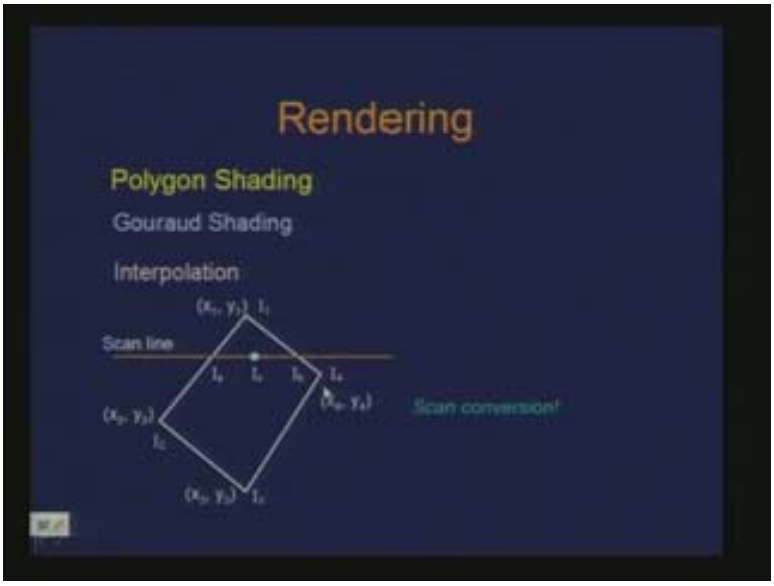
First of all we are talking about computation of vertex normal which is one essential component. We are given this polygonal structure and we know how to compute the normal of the face or the polygon. Then the question is, can I compute the normal of the vertex given the normal of these faces. Therefore you can compute normal at the vertex

just as average of the normal of the faces incident at the vertex, this is a very simple method. So, V for instance is nothing but just an average of the normals for the faces which is incident at the vertex. In fact this is the situation when I have the definition of the surface as collection of polygons or faces. I could very well have the definition of the surface in a form where I can explicitly or analytical compute the normal at that point just by the definition. That could be another way of computing the normal.

(Refer Slide Time: 00:22:35)

If I already have the definition of the normal at a point of the surface then I would rather use that because that is more accurate. Again looking back at the example of the tea pot these are the normals which we are referring to as, the normals at these vertices. Hence once these normals are available and once they have been computed then I can do this interpolation. Now that I can compute the normal at each vertex of the polygon I can apply the illumination model to compute the intensity at these points .1 .2 .3 and .4 I can compute I_1, I_2, I_3 and I_4 for these points.

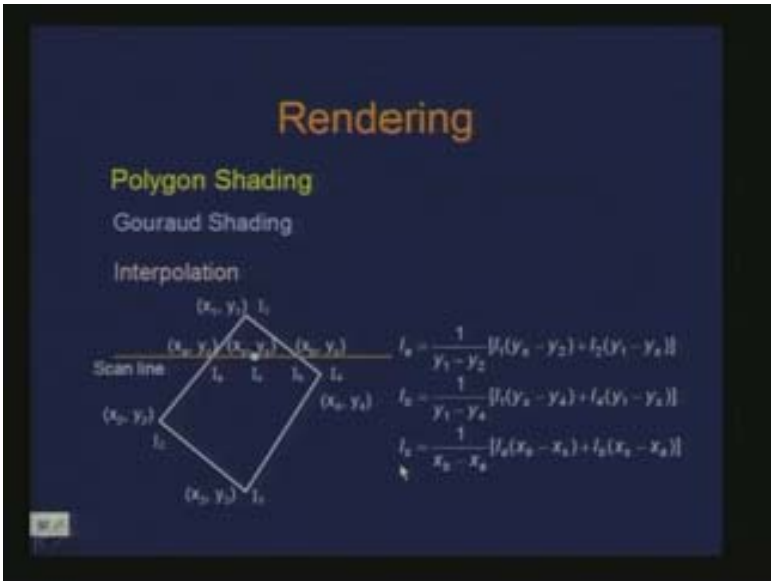
(Refer Slide Time: 00:24:49)



Now it becomes a matter of filling this polygon using interpolation through the intensity which has been computed at these vertices and that is something we have looked at in a very similar way as the scan conversion where we wanted to fill the polygon using scanline conversion. So one can in fact employ a similar approach here and that approach is, if this point is located at $x_1y_1$and this at x_2y_2, x_3y_3 and x_4y_4 and this is the current scanline so all I need to have as information is this intensity, this intensity and then I can perform a linear interpolation to get the intensity at this point which is I_s. So this intensity could be computed from linear interpolation from here and here I_1 and I_2, I_B can be computed using linear interpolation I_1 and I_4 and I_S can be computed using cleaner interpolation of I_A and I_B.

Basically the pixels which are intercepted here needs to be rendered. So, if I have a different scanline then I am talking about different pixels. Here if you have in the definition of this point given as X_{ay} as, this point as X_{by} as and this point X_{xy} as then you can compute the intensities I_A, I_B and I_S just using linear interpolation.

(Refer Slide Time: 00:27:00)



So here if I have the horizontal axes as X and the vertical axes as Y then my interpolation parameter T kind can be just Y.

(Refer Slide Time: 00:28:50)

Although strictly speaking this could be the distance along X, I can very well use Y using similar triangles. So, basically I can compute the intensity at the point $X_s Y_s$ given by IS which is nothing but two pass linear interpolation. One is to get these intensities and the other is to get this. This is the sort of example you will get for the same tea pot, for the same number of polygons. So clearly when we see here the shade of the object it is smooth. Is there a problem or any limitation with this gouraud shading? You have to do

computation for every point on the polygon which is the rasterization of the polygon. I am asking more in terms of the features which it can or cannot capture properly. What happens to specular reflection? How would it handle it?

One of the problems would be, if I have a triangle and what I observe is a specularity which would occur in the center of the polygon.

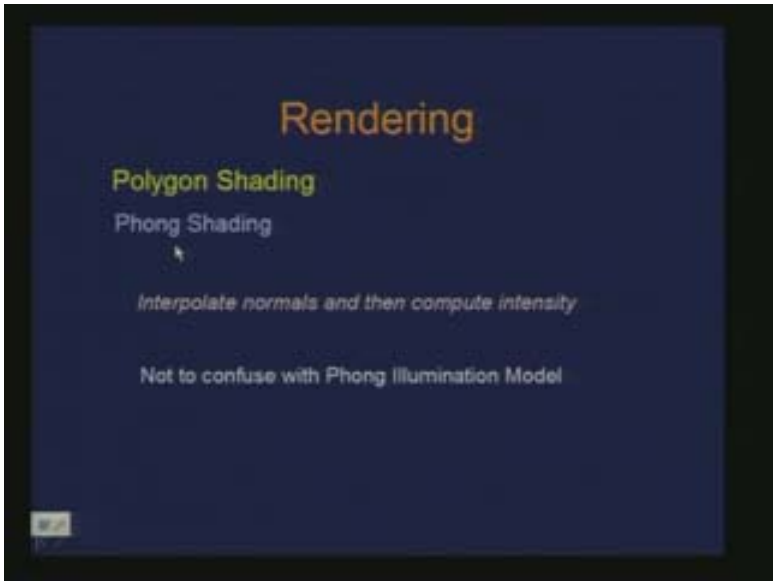
(Refer Slide Time: 00:30:50)

The computation at each of the vertices of the intensity would not capture this. So I am going to this miss out. Hence the rendering of this polygon would not have any specular components whereas it should have.

Similarly, if I have different specularity for vertices so here there is a large specular component and these are small specular components so the intensity which is computed here the aggregated intensity which has a large specular components could get spreader over because of the interpolation so I may not have that concentration effect of the shininess which was perhaps to have only at the small neighborhood of the vertex. So the specular reflections or the highlights are not properly handled. Even the Mack band effect which we looked at the defect is perceived less compared to the flat shade but it is still perceived and it does not get illuminated.

Therefore, in order to overcome this limitation where we can handle specular reflections properly what is done is that instead of interpolating the intensities which get computed at the vertices what one can do is interpolate normals. Therefore in some sense you compute normals at each point and then compute intensity at that point. This is called as phong shading.

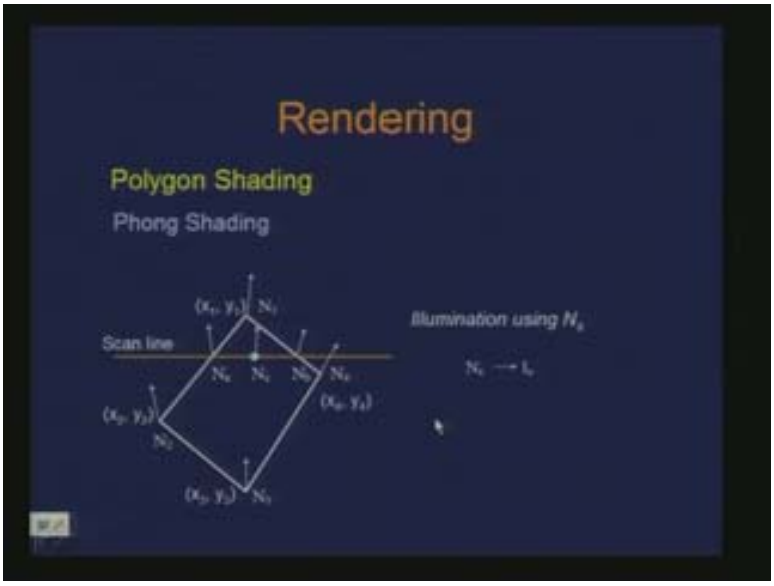
(Refer Slide Time: 00:32:24)



Phong shading is basically a polygon shading model which is different form phong illumination model. As in the case of gouraud shading here you have normals at each of these points.

In a similar manner as we did for gouraud shading for each scanline we compute or we interpolate the normals to get N_a and N_b and again we do the second pass of interpolation to get the normal at this point which is N_s. Then we compute illumination using this normal at this point. Since we are going to conduct the computation of illumination at each point the normal computed in this manner we would not have the problem of specular reflections.

(Refer Slide Time: 33:18)



Features of the phong shading:
It is more accurate dealing with specular components and the Mack band effect also turns out to be reduced and in some sense when we are doing this interpolation of normal we are trying to achieve some sort of an approximation of the shape. Normal is a shape indicator so when I am computing the normal at each point in some sense I am approximating the shape.

(Refer Slide Time: 00:35:18)

For example, if I have an original surface of this kind which is decomposed into polygons as this is one polygon, this is one polygon and this is another polygon and this is the

normal at this point N_2 is the normal at this point so the intermediate normals which I am computing using interpolation are in some sense capturing the shape of the original surface. So computation for each point in some sense is a representation of the surface which I have. Though it is an approximation at least it is an approximation of the surface. Therefore one of the disadvantages of phong shading is clearly the computational effort because you have to compute normal at each point then compute illumination at each point so the computation involved is more. This is the example for the same object where you see the specularity in a different way.

(Refer Slide Time: 00:36:35)

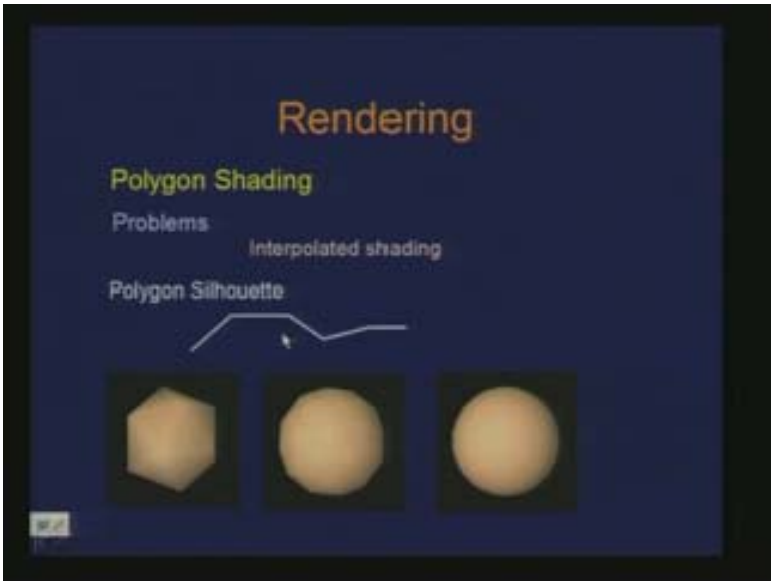
You see some concentration of shininess which in the case of gouraud shading gets spread around. In a little more detail if you see this is the flat shade we had, this is the gouraud shad we had and this is the phong shade. There are still problems as one indicated that depending on how you scan this might change. There are two light sources.

(Refer Slide Time: 00:35:38)

Here if you keep dividing the object into final and final and final polygons then eventually you will capture the components because the intensity computed for the polygon or for each of those vertices would in some sense be very similar. Hence within that there will not be any variation which should be different from what you compute like specular components at each of the vertex. That is what are trying to avoid in some sense, we do not want to have that final [d….] or decomposition of the object. We still want to have the polygonal representation in order to handle computation. This case is particularly pertaining to the fact that we are using interpolation. The fact that it is an interpolated shading there are some inherent problems.

One of the problems which are of course independent of anything we have is the fact that we are using polygonal representation of the object. This silhoutte which are the edges of the boundary as we see are always visible.

(Refer Slide Time: 00:41:29)

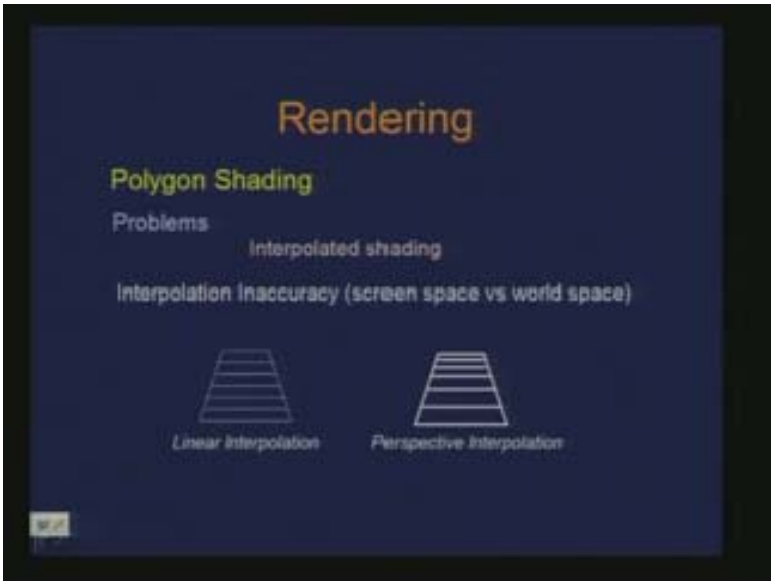


No matter what kind of shading you do for the interior part of the object this boundary is visible. Therefore it does indicate that the object is polygonal. For instance, if I have a very coarse polygonal representation of the spherical of object this is what I would see and this is shows that it is a polygonal representation where there are sharp changes. And in fact if you see here carefully these are the boundaries and that is where you still see the Mack band effect.

Now if I increase the number of polygons to be used I get better results. So if I change the resolution of the polygons I have more and more number of polygons and the object looks smoother. There is also an effect of what resolution I choose to be able to have the object look smooth and realistic. The other thing is that there when I am using interpolated shading, we are actually doing an interpolation like a scan conversion in the screen space or raster space in order to decide the shade of the pixel.

Basically on what we are trying to do interpolation is particularly phong shading and the entities are defined the world space normals and the vectors are defined in world. Now what is happening is that I use linear interpolation and I am performing this linear interpolation on the screen space with an intention that I do this linear interpolation in the entities of my world.

(Refer Slide Time: 00:44:20)

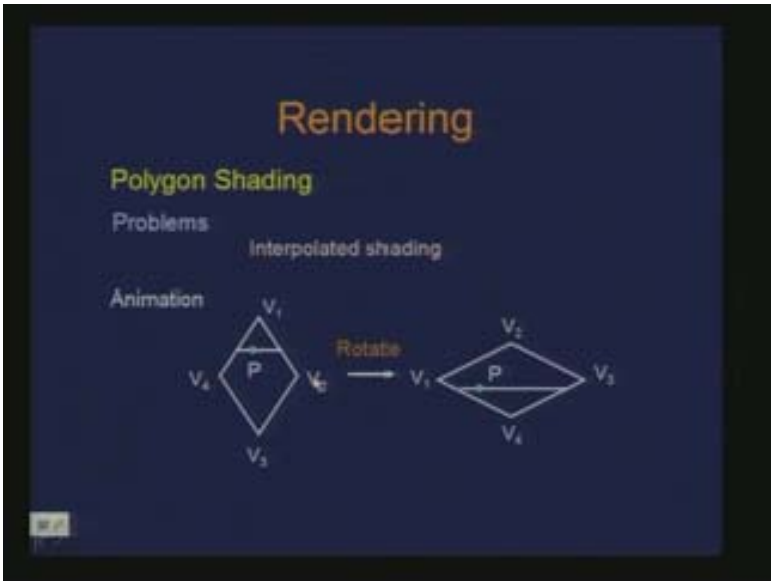


Hence, when I am using the equal inter scan distance due to the perspective transformation or distortion what I am actually doing is something like this. So if I want to achieve a linear interpolation in the true sense I need to do something like this, a counting for the perspective distortion or the foreshortening. In other words, if I have my horizontal axis as x and my vertical axis as y the equal change in y may not necessarily lead to the equal change in z which is coming after the projection then there will be a sharp change in z. So this is inherent due to the fact that I am performing an interpolation in one space of the entities which are defined in another space. And there is a non linear perspective transformation from this space to this space. This is another kind of a problem.

The other problem is something which relates to what was already mentioned there. It is the fact that if I am trying to compute intensity at a particular point so what happens when I move that object or do an animation?

Or, if we take a particular example of rotation I have the polygon defined as V_1, V_2, V_3 and V_4 this is the point I consider for defining or computing the intensity P. Now I rotate this so this V_1 comes here, this is V_1, V_4, V_3, V_2. I rotate this, now what happens? This point P which is the same point here would be computed through scanline which is this which is between the edges V_1 V_4 and V_3 V_4 instead of V_1 V_4 and V_1 V_2 that would give a different illumination or the intensity which is computed.

(Refer Slide Time: 00:48:35)

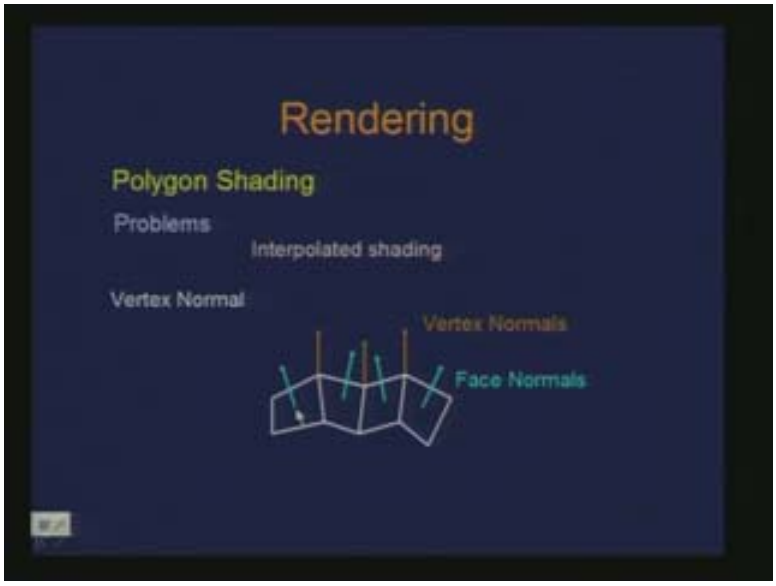


Here we are saying that for this computed intensity if I am not changing anything for the light source, you are rotating the object. If I am just rotating the object orthogonal to the light source why would there be a change. So unless I am changing with respect to the light source if I am moving away from the light source then I may want the intensity which is computed for the point to be different.

But here we are actually using the intensity computation for these vertices and everything else is ignored. Then at the end I should have this intensity to be the same. So just because of the fact that I am using this interpolation between the edges of the polygon this will turn out to be different. This is similar to if I scan horizontally or vertically it is the same argument. If I do a scanning like this then I talking about computing the intensity from different edges of the polygon.

Again there is an interesting observation. At one instance we are saying that these normals are trying to capture the shape of the object. So if I am doing this computation of the vertex normal where I am giving the face normal.

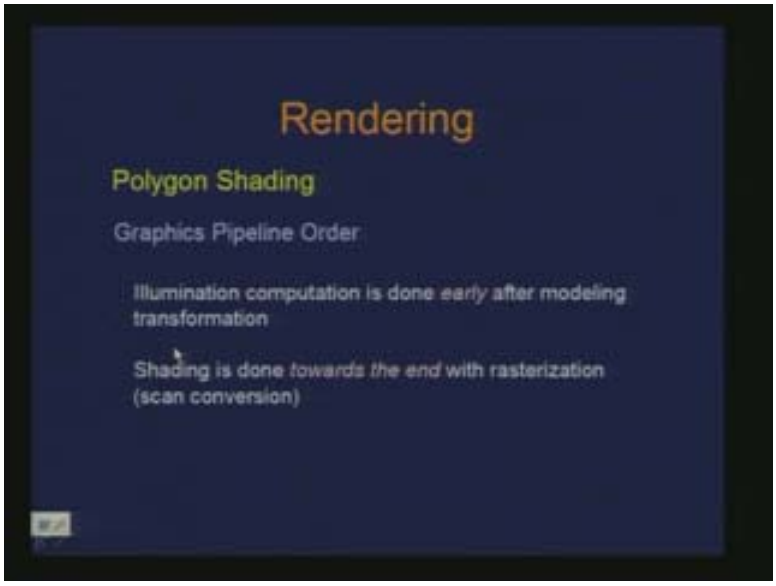
(Refer Slide Time: 00:50:47)



So this is face normal, this is face normal so the green ones are face normal. This is the kind of surface I have it is like a valley here and again a ridge there. When I compute the vertex normal at this point and this point and this point what you observe is that it turns out to be the same, at the same direction which is an indication to the fact that the surface is flat and which is not the case. When you do the computation of the intensity what you observe is that this whole thing is rendered in a flat manner because this normal is this, this is another normal and they are all the same. So there is a problem there too. You can resolve this also if you try do compute the normals only in the neighborhood or so there is some kind of a strip, you define this band at the border and that may take care.

If you look at from the point of view that how computation is defined in the graphics pipeline what we have is the illumination computation is done much earlier right after the modeling transformation whereas shading is done towards the end then we do the rasterization or scan conversion.

(Refer Slide Time: 00:51:36)



This is how these computations are located with respect to the rendering pipeline.