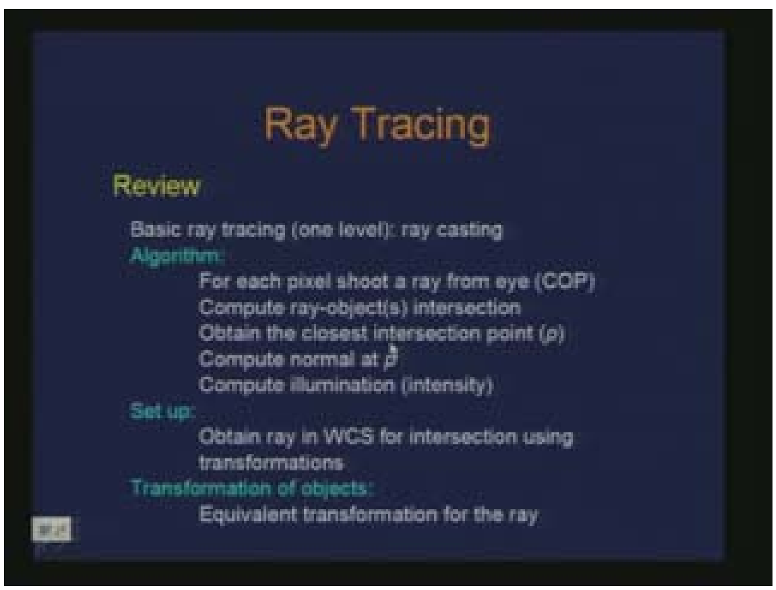
Introduction to Computer Graphics
Dr. Prem Kalra
Department of Computer Science and Engineering
Indian Institute of Technology, Delhi
Lecture - 26
Ray Tracing (Contd…..)

We will continue on ray tracing and just do some review of what we have seen so far. What we have basically looked at is a basic ray tracing method which is a one level ray tracing which is also called as ray casting where we are looking only at the intersection for the first hit closest object and subsequently we do not spawn the other secondary rays so there are only primary rays which we consider. This is what basic ray tracing is. If you look at the algorithm for this basic rays tracing would be that for each pixel (x, y) you will shoot a ray coming from the eye or the center projection for the camera which is the origin of your ray in the direction of the pixel (x, y). Now with that ray you compute ray object intersection the various objects in the scene then obtain the closest intersection point because that is the point which is visible and the others are occluded by the front objects.

(Refer Slide Time: 06:16)



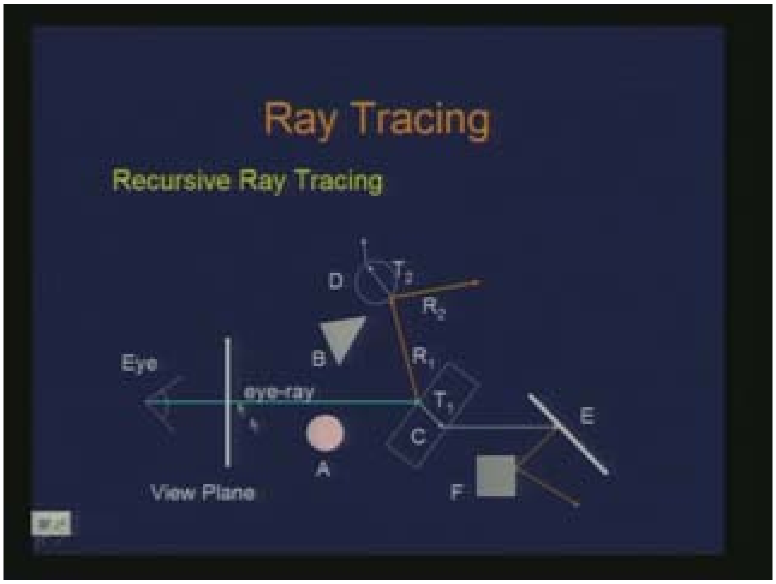
Once you have found out the intersection point P then you compute normal at that point because that is how you are going to compute the illumination for that point. So once you know the normal you compute the illumination using phong elimination model. That gives the intensity of the color in whichever way you want to render the image. That is the basic algorithm which is used. We also looked at some sort of a set up for ray tracing incorporating the user specification or the interface which will be given through the user in confirmation to what we have seen in the 3D viewing pipeline. So the basic idea is that you would like to obtain ray in the world coordinates system whereas the specification for the origin of the eye could be in viewing coordinate system.

So all it requires is set of transformations which would give you the ray in world coordinate system so that you can perform the intersection with the objects in the world. We also looked at the set up for ray tracing and other issues related to affine ray transformation. Therefore when I say objects we are basically dealing with the kind of primitives we have seen such as the sphere, the quadrics, the plane, box and so on for which we know the intersection and how to compute the intersection.

Hence if those objects are transformed using affine transformation then we would still like to use the same intersection routine with the generic primitives which are not transformed generally a non transform generic primitive. Basically that requires you to have an equivalent transformation which gets applied to the ray. So it is the ray which is transformed and then you compute the ray intersection in the generic set up. Generic set up means like you have defined the primitives like unit sphere at the origin and so on and those are the primitives. Therefore you make that transformation to the ray. Similarly you need a transformation for the normal at the point where you get the intersection.

Issues of ray tracing beyond one level:
Let us also look at the secondary rays. If you recall what we have as the scenario for ray tracing is that the ray starts at this eye or the camera center and this is your viewing plane.

(Refer Slide Time: 09:24)



So with respect to pixel on this viewing plane you get a ray which we call as eye ray and that is the primary ray which is spawned with respect to every pixel on the viewing plane. It hits an object C, there could be various possibilities and one possibility is that you get a reflected ray at this point which is R1 because the material of the surface of this object C is reflective so you get a reflected ray and you may also get a transmitted draw or transmission ray which is this steamer because the object is transmissive. Hence there is a transmission which can happen to the object.

Further this reflected ray may hit another object D and similar phenomena occur. That means another ray is spawn as a reflected ray R2 and a transmitted ray as t two from here and same thing may happen here. So here we are considering the transmitted ray through this which may hit another object E and so on. Now, in order to compute the illumination at this point that is where the ray had hit the first. That is, the point which you want to illuminate in order to able to do so you will have to account for all these reflected rays and the transmitted rays which are spawned as a consequence of this. Therefore you will have to trace these rays and accumulate the illumination they would contribute to this point which will be the final illumination of this point.

Let us look at the configuration which gets built using the various rays we have. So all we observe is actually a formation of a tree.

(Refer Slide Time: 11:12)

This is the eye so what we get is the eye ray here which is the primary ray, it hits the object C here and there are possibilities that you have a reflected ray R_1, transmitted ray T_1 which in turn may hit another object D and it also has a reflected ray R_2 and transmitted ray T_2 and so on. A similar thing can be happen here to the transmitted ray. Now what we are looking at is if you want to compute the intensity at the point of intersection with the object C what you need to account for is what happens with respect to the object D and what happens with respect to this object and so on.

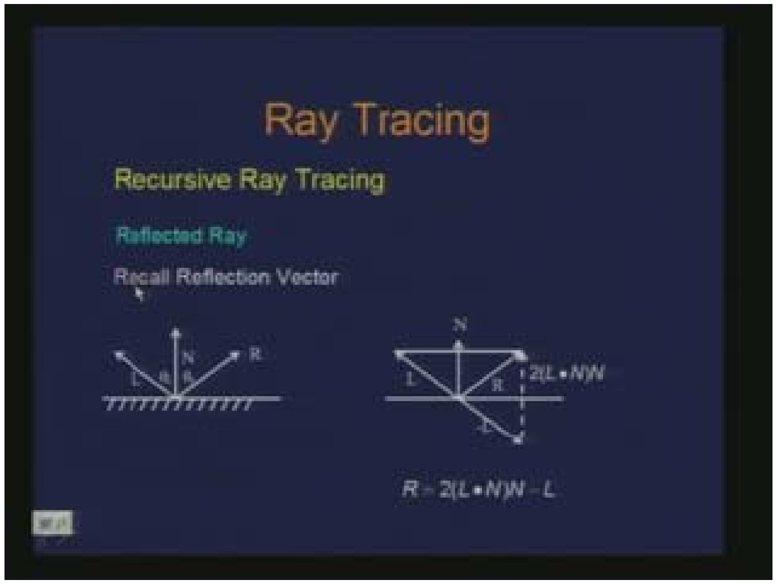
So in some sense you will have to count for the illumination at this object, this object and then compute the total illumination at this point. And as a computation you can see that this is a just recursive formulation, at the point hit you spawn the reflected ray and the transmitted ray and that may be done in a recursive form.

Now if you look at the trace in addition to the primary rays which we have been talking about the eye ray is the primary ray which goes through the particular pixel you want to eliminate through the eye. This is the eye ray, it hits the object and there are various other rays now which one has to look into. This is the reflected ray, this is the transmitted ray and there is also a ray which you have to look at which is the shadow ray. Shadow ray is just the ray between the point which is intersected or hits through the light source just to figure out whether this point is occluded by any other object with respect to this light source or not.

(Refer Slide Time: 12:44)

If it is so then this point is in shadow. So these are the three secondary rays which one has to consider when you find out the point of intersection. The next thing is that, when you are looking at various rays we also have to account for what happens with respect to these secondary rays then we get to know what the direction of this ray is and only then you can trace them.

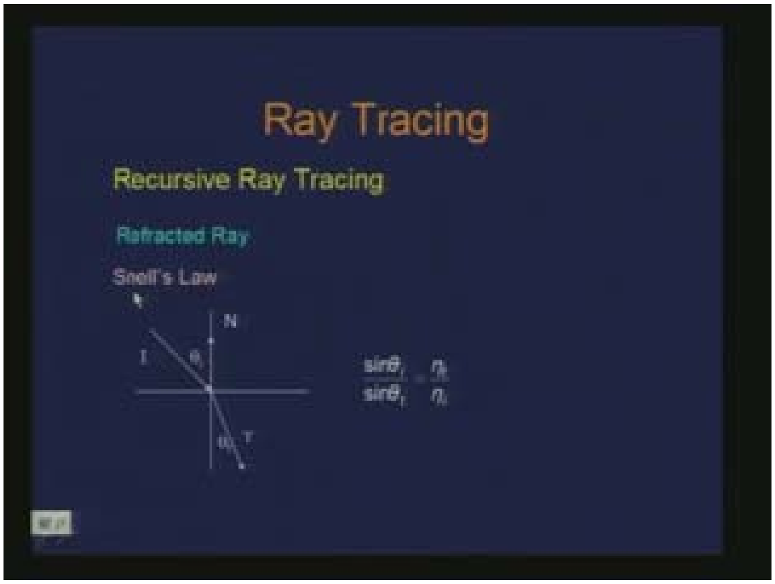
(Refer Slide Time: 15:17)



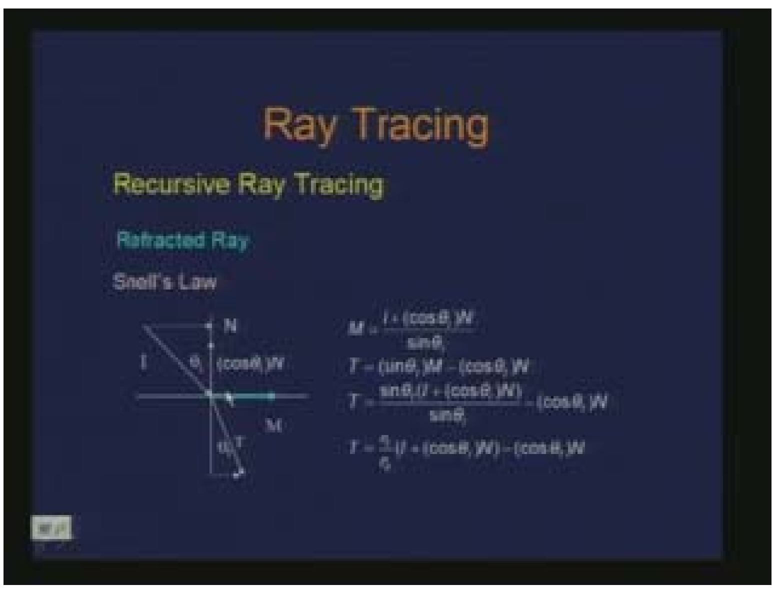
So, there is a reflected ray, there is a transmitted ray and there is a shadow ray. So, for the reflected ray if you recall how to compute the reflection vector that is what we need to apply here. So, the reflection vector basically follows the law of reflection where if you have this, so this was in the context of with respect to the light vector. So this is the light vector, this is your normal vector, then this r which is the reflected vector is obtained just by considering the fact that this angel theta eye is equal to the angle theta r. And geometrically you can figure out that this r is nothing but this which is 2(I dot N) N minus L. So in the context of this, this L is nothing but in the first case the eye ray and with respect that you are going to compute the reflection vector. So the direction of L may be just he opposite what we are looking at. You have to incorporate the proper sign with respect to the direction you have for the incident ray and compute the reflected vector or the reflected ray. This is how you get the reflected ray.

Let us see how to get the transmitted ray. So if you recall Snell's law of high school physics we have the incident ray here and depending on the type of medium it is incident on you will observe a bend of the ray and that is what the direction of the transmitted ray would be governed by the refractive indices of the material of the medium from here to here. So we observe this relation sin theta i times theta i ni is equal to sin theta t(theta t) by Snell's law.

(Refer Slide Time: 16:29)



(Refer Slide Time: 20:33)

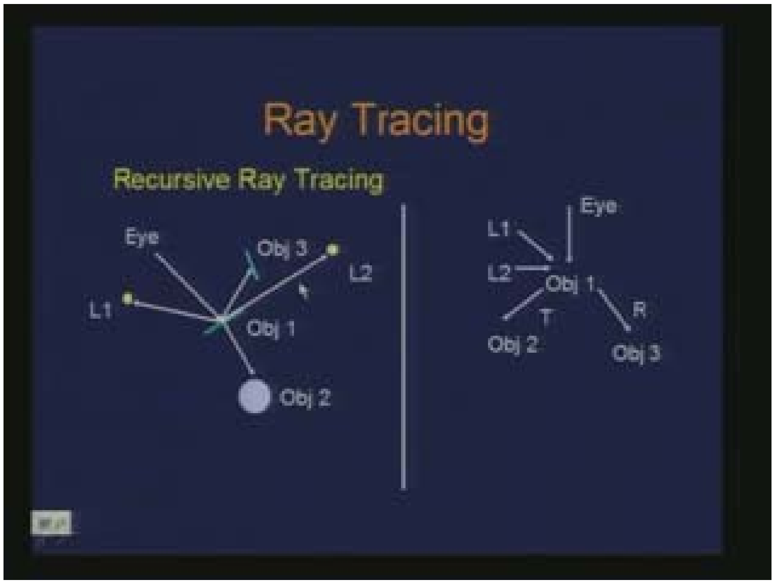


So we are interested in finding out the direction of this T so that we can continue tracing the ray. So similar to what we have done in the case of reflected ray we can actually do a geometric construction and obtain this T. What we are trying to do here is that this is the incident ray I have, this is theta i and this angle is theta T and this is the direction of the normal I have. First we try to obtain the unit vector in this direction. So if I consider this triangle then with respect to a unit vector in the direction of the incident ray I have a vector which is cos theta i in the direction N. Then this vector is nothing but i plus this vector which is cos theta i(N) and if I am interested in the unit vector then it is divided by the magnitude of this vector which is sin theta i. So I obtain a vector M which is a unit vector in this direction shown in red color here, this M vector is basically given as this i plus cos theta iN by sin theta i. That is the unit vector I get here.

Look at this triangle now. I am basically interested in finding out T. Now again T is from this triangle which is this vector plus this vector. This vector is nothing but the unit vector times the magnitude here. The unit vector is M and sin theta T is the magnitude if I consider this to be a unit vector here. So I have this vector as sin theta T(M) and this is nothing but cos theta t there is an opposite direction N so I have minus cos theta T(N). So the summation of these two vectors gives me the vector T. Now I can substitute for m which I have already computed there to get this and I know the ratios of sin theta t to sin theta i which are nothing but the reflective indices which is this and I get i plus cos theta N minus cos theta T so here also cos theta i can also be computed as the dot product of I and N incorporating the proper signs. This is how I can get t which is the direction for the transmitted ray. Now what we have is the reflected ray direction and we also have the transmitted ray direction.

It is just a matter of tracing the respective rays. Therefore I have the objects as object 1, object 2 and object 3 and these are the light sources L_1 and L_2 which is my eye. This is my primary ray i ray which starts from the eye to a pixel (x, y) and hits the object 1 here somewhere. So with respect to the tree which gets formed I have this ray coming to object 1. In the next step what happens is at this point I have a reflected ray here which goes and hits the object 3, I have a transmitted ray which comes and hits the object 2 and then I have these shadow rays with respect to each of the light sources L_1 and L_2. This is the scenario which I have once I hit the object 1 so I have to look at all these rays.

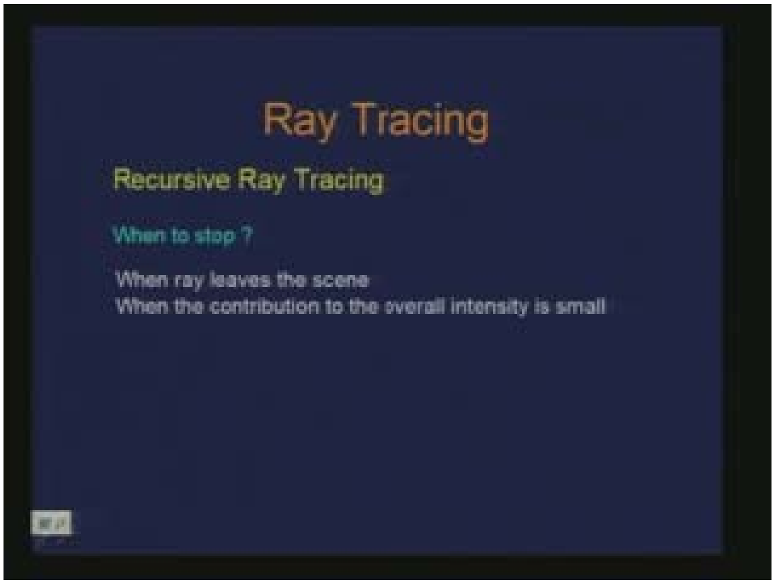
(Refer Slide Time: 24:02)



So here on this tree what I have is this is the object 1 so I have the transmitted ray t which goes and hits the object 2 and I have a reflected ray R which goes and hits the object 3. Then I have these rays L_1 and L_2. In the next iteration again at object 2 I have a reflected ray here and again I have these shadow rays with respect to L_1 and L_2. And similarly here with respect to object three I have this as the reflected ray and this is the transmitted ray and these are the shadow rays. So this is what you have to account for in this scenario. While calculating the illumination here I do the elimination computation with respect to L_1 and L_2 and then I look at what could possibly come from the other rays which are spawned and then I accumulate here.

The next thing is when to stop? That is the basic question. It may keep going on forever. The first thing is if we figure out that the ray leaves the scene that means it does not intersect the ray which you have, the secondary rays which are generated do not intersect further so they leave the scene so you can stop there. But that itself may be too much. For instance if you take a closed environment like a room so the ray might keep coming and going hitting the floor, ceiling, walls forever so your ray tracer will never finish.

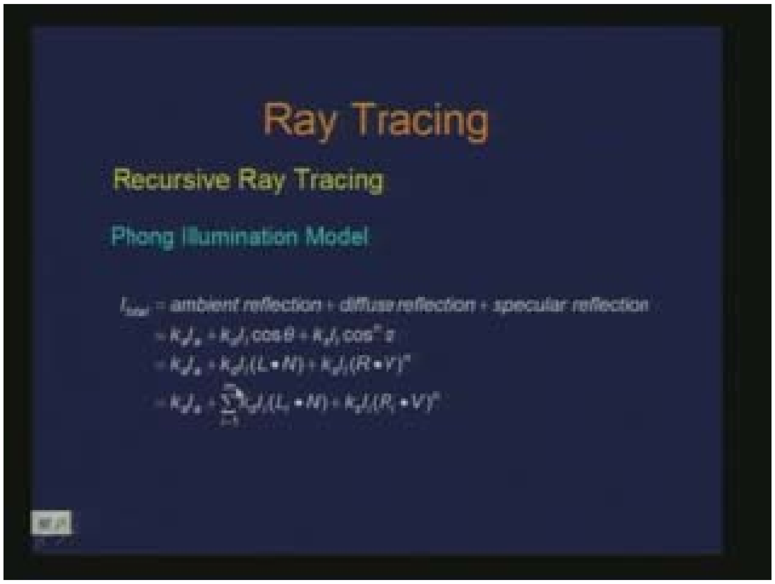
(Refer Slide Time: 25:41)



So the question is, what is it we are trying to use these rays for? Basically looking at the contribution of the secondary rays which get further spawned to the overall intensity at the point which was hit first there is some sort of an attenuation effect which you get through the respective transmitive or the reflective coefficients, you compute intensity and then you [we…] through these coefficients. If that contribution which is there of the intensity to be accumulated to further level up is very small then you can neglect it. After a certain depth of recursion you figure out that the contribution which is being made now is very small because of the attenuation which you have so you can just stop it there. The maximum depth of the tree you wanted could be defined by as a user and you stop it after that.

Again let us look at certain issues in terms of computing the intensity. When we look at the phong illumination model we have these components so the total intensity at a point has got three components the ambient component, the diffuse component and the specular component and these are the computations which are involved for having these components. And you can have multiple light sources. So, for the multiple light sources all you need is a summation from I to m where m is the number of light sources we have.

(Refer Slide Time: 29:50)



Now, if we are looking at the scenario where we have these secondary rays, if we have only one light source and we figure out that the point which was hit the point of intersection is in shadow. What we have got is the point which is intersected we figure out that it is under shadow.

(Refer Slide Time: 31:04)

Therefore what we do to the illumination? There is no contribution to the point through the light source. If I have this single light source then we are talking only about the ambient and this is when I have only single light source. So, if I have a point in shadow and if I am looking at only this situation where I have a single light source then the total intensity is nothing but the ambient reflection. And if there are multiple light sources you may still want to look at what happens with respect to other light sources. For simplicity many a times a ray is not spawned further if you find out that the point is under shadow. If you figure out that the point is under shadow you do not try to see whether there is a reflected ray and a transmitted ray and just render it using ambient intensity.

First of all if you will have multiple light sources you may still want to consider what happens with respect to other light sources because you may still have light coming from other sources. If you have a general situation where I have reflection and transmission rays then what we are looking at is the local illumination which means that all these three components the ambient component, the diffuse reflection component and the specular are component computed with respect to each of the light sources.

(Refer Slide Time: 34:19)

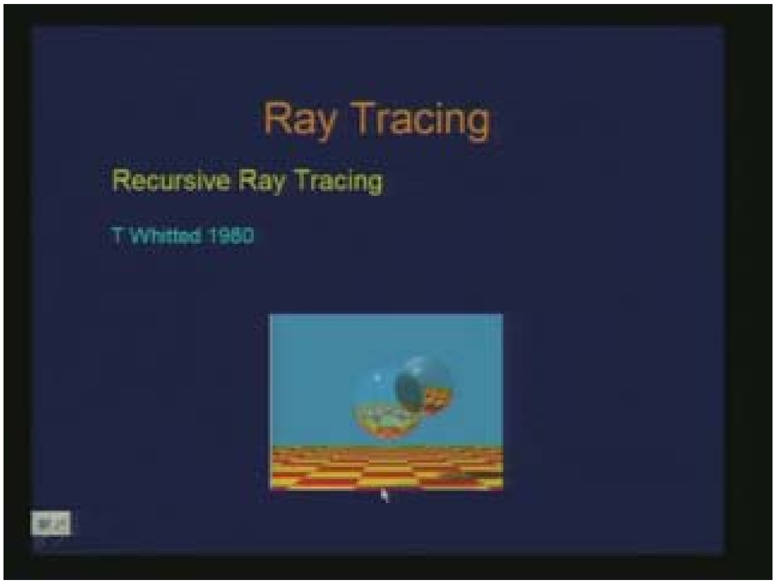
Therefore that gives me the local illumination at that point. Further when I am looking at reflection and transmission rays then I gather the illumination with respect to the point which is head by the reflected ray, the illumination for the point which is the head by the transmission ray and do an attenuation using the reflective coefficient and the transmitted coefficient.

Now, if there is further spawning of the rays in a recursive manner you see that this attenuation factor is going to reduce the total effect from the nodes which are lower in the tree. So when you are computing the contribution you are computing through these coefficients and that is where you can apply a certain limit where you want to stop now. So it is actually some sort of a limit to this because this is getting successively multiplied.

So, the next time there will be another K_{rg}, next time then another K_{rg} and so on. Therefore you can apply a limit just to this.

In some sense what is happening is we are capturing some notation of global illumination. And there is a local elimination which is the consequence of direct composite of elimination at the point with respect to the light sources. And there are also other contributions coming through the reflected and transmitted rays. In some sense we are having a notion of global illumination. This was first done by Turner Whitted back in 1980 and this is the image he rendered.

(Refer Slide Time: 35:32)



Of course this floor is just a pattern and these spheres are reflective and refractive so you observe some distortion of these here and you also have shadows here and shadows here so his idea was that ray tracing in some sense accounts for global illumination. However, later on the notion of global illumination was further refined. Basically you are accounting for an interaction with respect to other objects through reflections and transmissions.

Here is an example where no reflective ray or transmission ray was considered. It is just one level ray tracing with shadow rays. There are multiple light sources, there is a light source here which gives you shadow here and a shadow here and a shadow there but there are no reflections. So it is a one level ray tracing with shadows.

(Refer Slide Time: 38:38)

Here we have reflections. Here you see the reflection of the yellow thing on this, there is a reflection of the ceiling somewhere there which is in blue coming here so this is with reflection. By adding just one aspect it changes the realism. So I would basically recommend you to also look at the webpage of Barrack. He wrote a ray tracer as a part of the assignment in the graphics two course which is a complete ray tracing and not just the basic ray tracing. It also contains a reference to the POV- ray home page. POV-ray is another ray tracer which is available in the public domain.

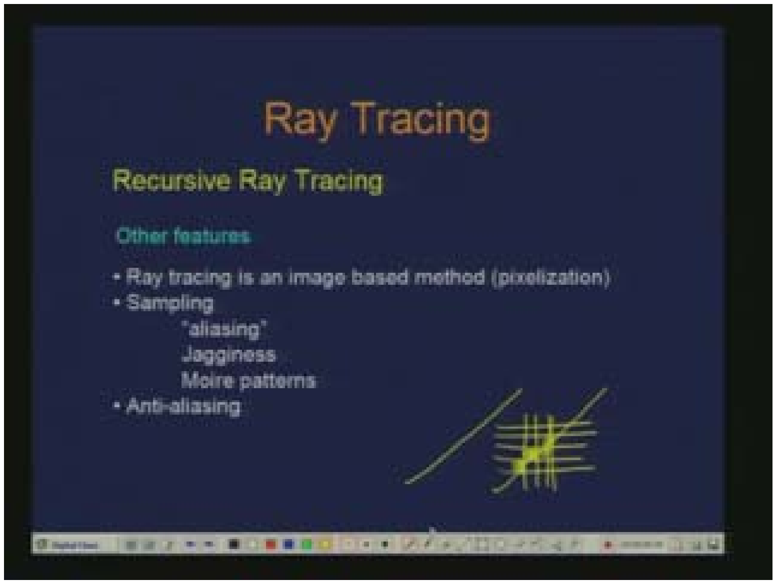
Other features which are of concern to ray tracing:
Here we had not considered any reflections. There is supposedly a ceiling which is of a blue kind of a color and this floor is yellow and what you see here is a reflection of this part to this part and from the above to this part. So what is happening is, here you see some sort of an accumulation of the shadow. This is a shadow with respect to two light sources so it is a union of two shadows in some sense in terms of the effect. That means it is not getting light from either of the light sources. The other features which are also pertinent is that, one thing is when we look at ray tracing as a method of rendering it is actually an image based method because what we are doing is we are shooting a ray with respect to each pixel and then perform intersections and so on and so forth.

(Refer Slide Time: 40:38)

So, the number of rays which you are going to use to generate an image depends on the fact that how many pixels you have for the area you want to render. Hence there is some sort of a pixelization or rasterization which is implicit there when we are performing ray tracing. So when I say pixelization or rasterization it inherently involves sampling. That means how many pixels I have, what is the resolution in terms of number of pixels etc. This is the sampling which is going on. So this in effect gives you what you call as the aliasing effect.

When we are doing this line drawing algorithm where a dot or a pixel was drawn or not drawn what would you observe is that when a line is drawn as dots or these square pixels will in effect give you some jagginess a stair kind of a pattern, it will not be a smooth line it will be jaggi like steps. That is also aliasing and that is also a consequence of the fact that we are doing sampling for using number of pixels.

(Refer Slide Time: 48:02)



These Moire patterns are something like a shift of wavy patterns so these are also due to aliasing. Some would have seen aliasing as an artifact due to the high frequency components being impersonated as low frequency components in the spectral domain. Here the visual effect what we are looking at is this aliasing which is causing the jagginess so the question comes as how we remove this effect. In the case of ray tracing also we are shooting a ray through the pixel and eventually I just color that pixel with respect to whatever happens to the intersection there.

I could have many points contributing to this pixel in the object precision so that would cause these artifacts so the methods which try to remove this artifact of aliasing are called as anti aliasing effect.

A pixel is nothing but some sort of a grid. so what you saying is that if the line passes like this you are going to render this pixel, may be this pixel, may be this pixel and may be this pixel so what you observe is this tough kind of a thing, it is not a smooth straight line but because of the fact that you are using these pixels to render you will see these square patterns and this is basically governed by the number of pixels you have for the area you are rendering.

If I have lots of pixels then these dots are going to be small and small so I would not see this artifact whereas if I have a small resolution less number of pixels for the same area I will see this effect extremely dominating. So similar thing is happening even in ray tracing because I am actually shooting rays with respect to these pixels only then I color that pixel. So this pattern would exist. Particularly it will be apparent at the boundaries at the two objects or boundaries at the shadows.

Now given this as an artifact how do you propose to reduce it if not eliminate it? What about virtually increasing the resolution? This means I do some sort of a super sampling.

Super sampling means that instead of using single ray per pixel I use more number of rays per pixel.

(Refer Slide Time: 54:11)

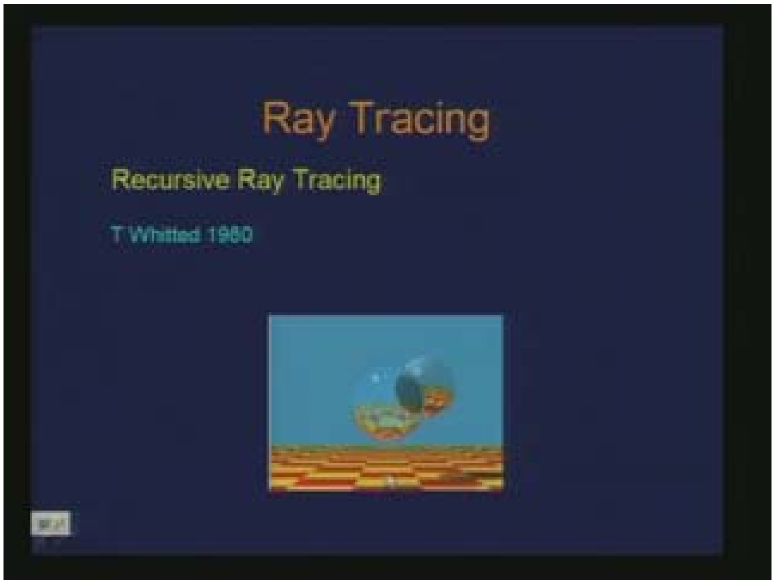
Then to get the final result I do some sort of an averaging or some other filter. Average is nothing but some sort of a smoothing function it is a filter. So, for instance if I had only one ray which was used for rendering this whole pixel instead of doing that now I use five rays in addition and look at the contribution of each of these rays in terms of the color or the intensity I obtain and do some sort of an average to render this whole thing. So I have increased the virtual resolution, the computation in which I am doing this ray tracing has got a higher solution than I had decided earlier.

When you look at the boundaries here you see some kind of a stair case pattern here, here, here and here. This is what is meant by aliasing. And once a super sampling is used you see that the effect is considerably reduced. The effect is there but it is much less than this. Basically we are doing this super sampling and doing this smoothening or blurring because of which we do not see these boundaries.

Then what is the consequence in terms of computation? The consequence is bad. For each ray we are trying to spawn five rays. This was the ray earlier. Now what I am doing is I am basically looking at these as further divided and spawn these rays. You can do some optimization to exploit the coherence information of these rays but at a brute force level you are adding computation just by the number of rays you are shooting now.
But the effect is quite good. The result what you get is quite good. in some cases what happens is instead of using only the position like this and that is where the problem comes of not been able to use the coherence is that even this is a very structured way of doing super sampling.

What you want is some randomness to it. You are doing a sampling so you do not want to do a structured sampling. You add some stochastic thing to it randomness to it and that is what we call as jitter. Therefore we jitter these pixel positions and then we shoot the rays. So these positions may not be retained at these points because the idea is that you sample as much as you can otherwise you will again see certain artifacts at the boundaries of these pixels so some jittering is done. The other issues with respect to ray tracing: One is in the context of texture mapping.

(Refer Slide Time: 55:14)



The image by Whitted is an ideal example for texture mapping. The image by Whitted was a pattern but you could also think this to be a texture on the floor so the question comes as how we map a certain pattern or a texture on the top of objects. Even if you want to write this pattern on this pen it could be done by using texture mapping. And it is very easy to incorporate texture mapping in ray tracing.