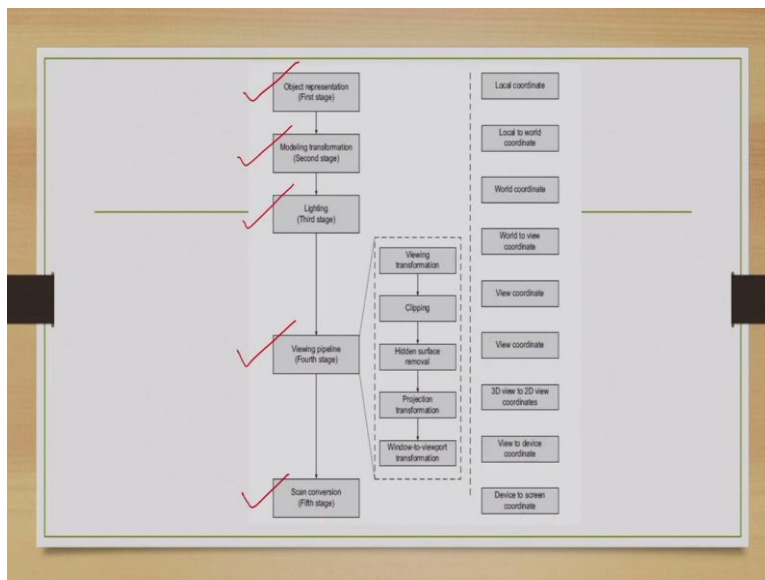


**Computer Graphics**  
**Professor Dr Samit Bhattacharya**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Guwahati**  
**Lecture No 16**  
**Intensity Mapping**

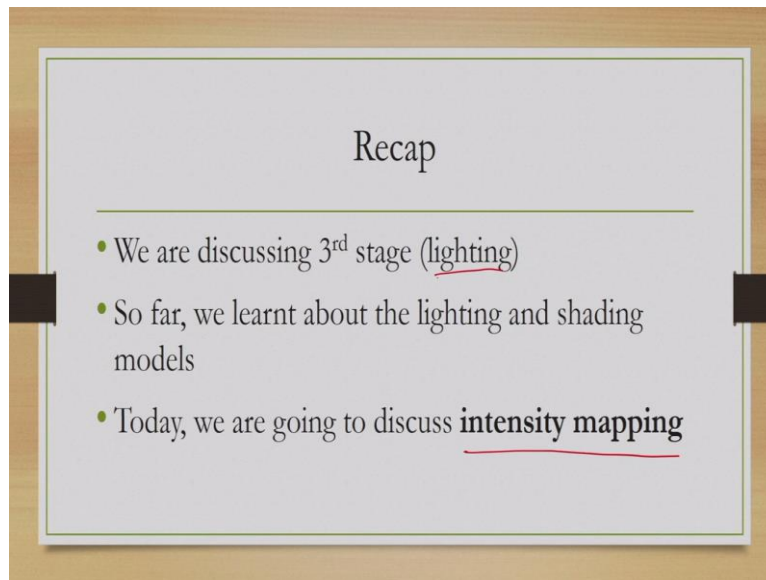
Hello and welcome to lecture number 16 in the course Computer Graphics, we are currently discussing the different pipeline stages, pipeline means how the rendering of a 2D image on a computer screen takes place through the process of Computer Graphics.

(Refer Slide Time: 00:57)



Now, as we know, there are five stages; we already have discussed the first two stages, namely Object representation, and Modeling transformers. Currently, we are discussing Lighting or the third stage and after this, we will be left with two more stages to discuss the fourth stage Viewing pipeline and the fifth stage Scan conversion.

(Refer Slide Time: 01:25)



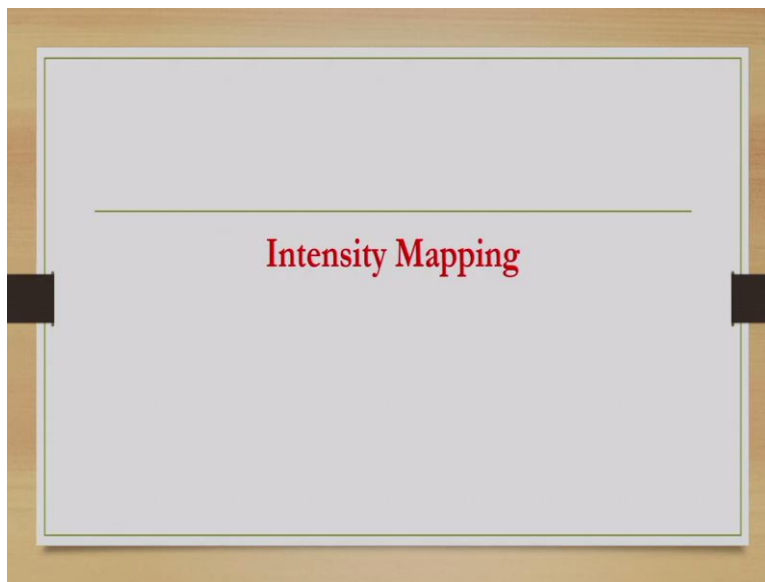
In the third stage Lighting, we deal with assigning colors to the surface points, the surface of an object. Now, in the previous couple of lectures, we have learned about the process of coloring that is, we learned about a simple Lighting model, and also we learned about Shading models. To recap, the Lighting model is a complex mathematical expression to compute color at a given point and it makes use of various components of lights that are there when we try to see a colored object.

Now, these components are ambient light, diffused reflection due to direct light source and specular reflection due to direct light source. And for each of these, we have learned models and these models in turn make use of the vectors, surface normal vectors or the viewing vector and the vector towards the light source, all these vectors are used to compute these components. And at the end, we sum up these three component contributions to get the overall color values which is expressed in terms of an intensity value.

Now, this Lighting model is complex and involves lots of operations. So, essentially it takes time in order to reduce computation time; we learnt about Shading models, where we do not compute color values using the Lighting model at each and every point, instead we compute values at a very small number of points, maybe a single point on a surface and we use interpolation techniques to assign colors to other points.

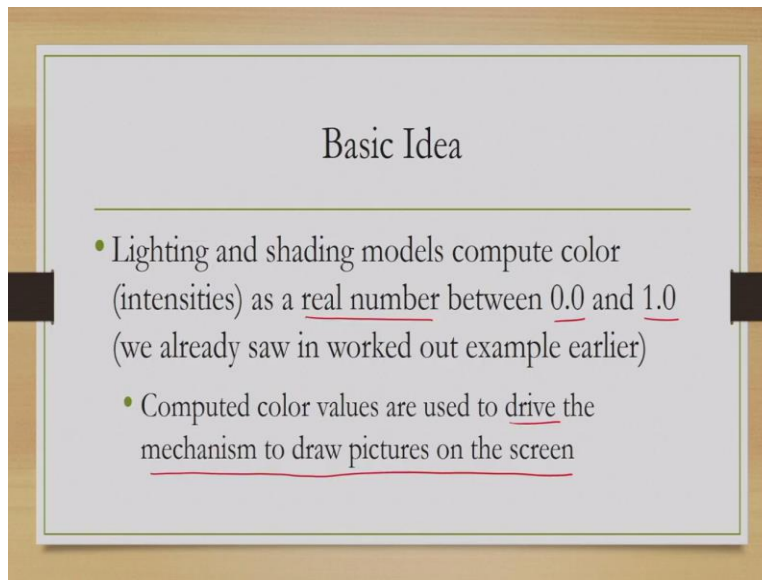
Now, this interpolation technique is much more simpler compared to the Lighting model computations. Now, these two techniques to assign colors are discussed in the previous lectures. One more thing remains that is how we map the computed intensity values either using the Lighting model or using the Shading models to a bit sequence, a sequence of 0's and 1's that the computer understands that will be the subject matter of today's discussion, Intensity Mapping. This is the third component of assigning color to an object surface.

(Refer Slide Time: 04:25)



Now, when we talk of Intensity Mapping, what we refer to? We refer to a mapping process, what it maps? It maps the intensity value that we have computed using the Lighting or the Shading model to a value that a computer understands that is a string of 0's and 1's.

(Refer Slide Time: 04:50)



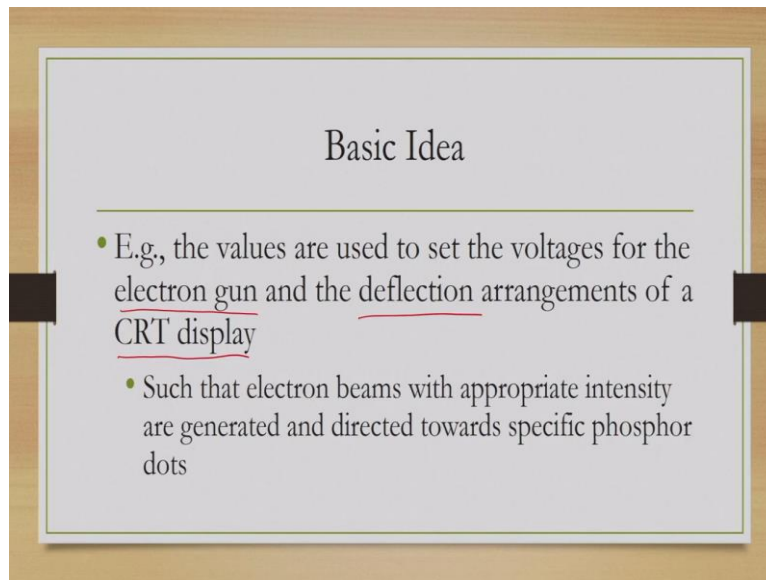
If you may recollect, during the worked-out examples that we have discussed in the previous lectures, we have seen the computation of intensity values. And those values are real numbers typically within the range 0 to 1. Now, these values are supposed to be used to drive the mechanism to draw pictures on the screen.

In the introductory lectures, we have touched upon the basic idea of a graphics system. There we mentioned that through the pipeline stages we compute intensity values and these values are used to basically drive some electromechanical arrangement which is responsible for rendering or displaying a colored object on a computer screen.

As an example, we briefly touched upon the idea of cathode ray tube displays. So, if you may recollect, there what we said that the CRT displays consists of an electromechanical arrangement where there are electron beams generated which are supposed to hit some locations on the screen representing the pixel grid. Now, this generation of electron beams is done through an electromechanical arrangement consisting of cathodes and anodes and magnetic fields.

And this electromechanical arrangement is controlled by the values that we compute at the end of the pipeline stages. So, our ultimate objective is to use the values, intensity values and use them to drive the mechanism that actually is responsible for drawing colors on the screen or drawing pictures on the screen.

(Refer Slide Time: 07:18)



As we have already mentioned, in a CRT display, this picture drawing is done by an arrangement of electron guns, which emits electron beams, and there is a mechanism to deflect those beams to specific regions on the screen where phosphor dots are present. And when the beam hits the phosphor dots, the dots emit photons with particular intensity that is light intensity, which gives us the sensation of a colored image on a screen.

Of course, CRT displays are now obsolete. You may not be knowing about these displays nowadays, but there are lessons to learn from CRT displays. And at the end of this course, towards the end, we will learn about other displays where similar things happen, where we actually use the computed intensities to generate some effect on the screen which gives us a sensation of color. And this computed intensity values are used to drive the mechanism that generates those effects. We will talk about some display mechanisms at the end of this course, where we'll have dedicated lectures on Graphic Hardware.

(Refer Slide Time: 09:00)

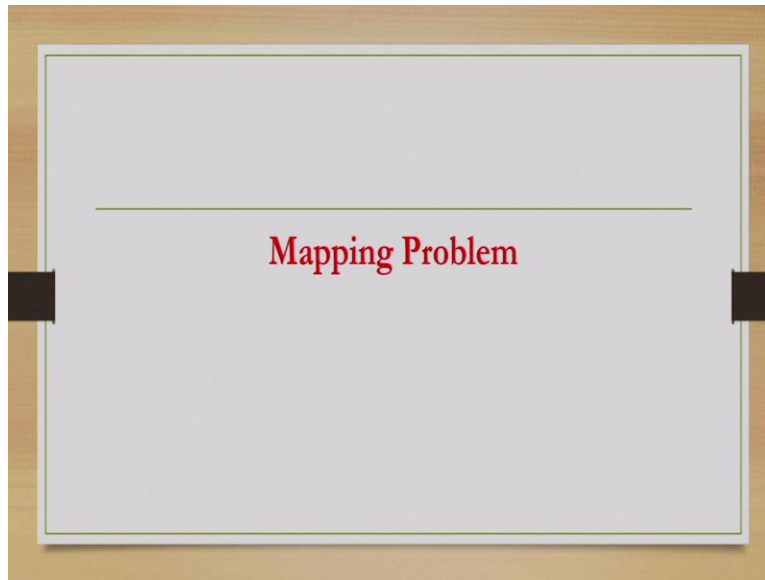
## Basic Idea

- Due to discrete nature of a computer, any real intensity value cannot be represented and used for the purpose
- Representation depends on frame buffer design

Now, the point is, so, we are saying that this intensity values are supposed to drive a mechanism some arrangement which in turn is responsible for generating the effect of colored image. But if the intensity values are computed as a real number in a range of 0 to 1, how we make the computer understand the value because computers do not understand these real numbers they only understand digital values, binary strings of 0's and 1's.

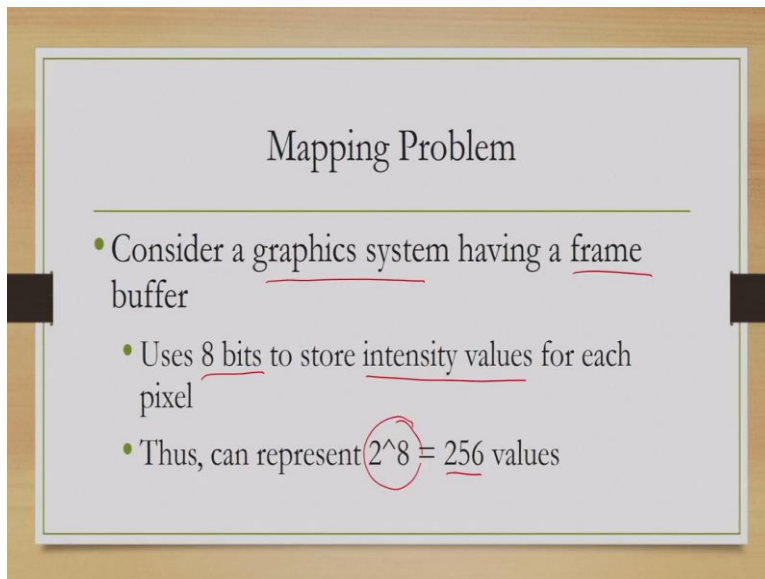
A problem here is that any intensity value cannot be represented and used for the purpose of driving some arrangement to generate the visual effect of colored image on a screen and we need some way to represent the corresponding intensity values in the computer. Now, this presentation depends on the frame buffer how we designed the frame buffer.

(Refer Slide Time: 10:25)



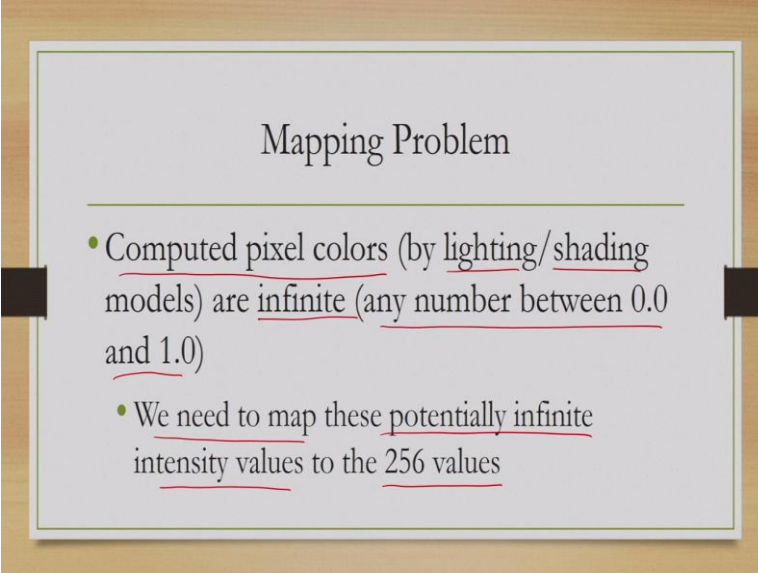
And that is what we call the Mapping Problem. What is this problem?

(Refer Slide Time: 10:40)



Suppose, let us try to understand it in terms of an example, suppose, we have a graphics system which has a frame buffer where 8 bits are there for each pixel location that means, 8 bits are there to store intensity values for each pixel. Now, with 8 bits, how many colors we can represent that is 2 to the power 8 or 256 values, it means that for each pixel location we can assign any one of the 256 values as a color value. So, for that particular graphics device, we can say that any pixel can take at most 256 color values.

(Refer Slide Time: 11:37)



### Mapping Problem

- Computed pixel colors (by lighting/shading models) are infinite (any number between 0.0 and 1.0)
- We need to map these potentially infinite intensity values to the 256 values

On the other hand, when we are computing the pixel colors, there is no such restriction, we can compute any value, any number between 0 to 1. So, that is essentially an infinite range of values. Note that this computation takes place with the help of Lighting or Shading models. So, on the one hand we have values that can be anything, which we get by applying the Lighting or Shading models real value between 0 to 1.

And on the other hand due to the particular hardware design, we can represent at most a restricted number of values for each pixel location, in our example it is 256 values. So, essentially we need to map this potentially infinite intensity values to the 256 values, this is the problem. So, given the set size, the size of the number of values that can be represented in a computer, we have to map the potential range of values to those restricted sets.



(Refer Slide Time: 13:03)

## Mapping Problem

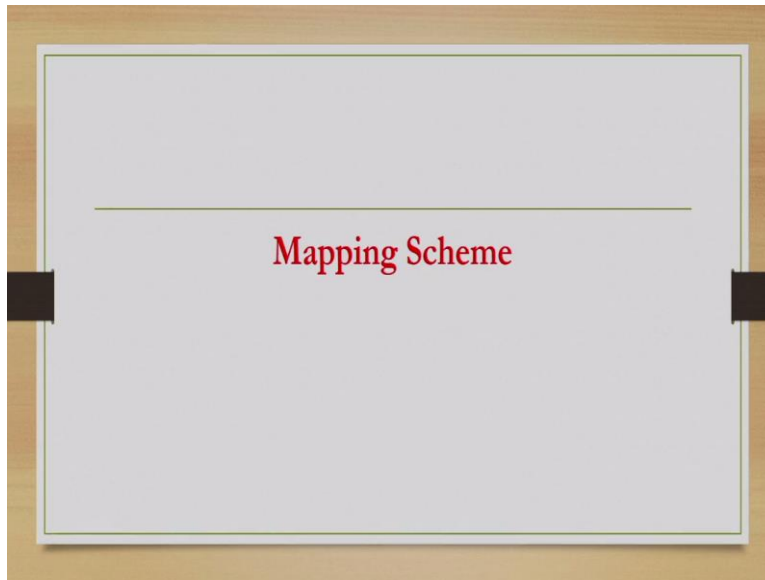
---

- We cannot use any arbitrary mapping,
- May lead to visible distortion in the image
  
- How can we achieve the objective?

This is our mapping problem, where we have to keep in mind that we cannot use any arbitrary mapping because that may lead to visible distortion, our perception is a very sensitive and complex thing. If we arbitrarily decide the mapping, then we may perceive images in a different way than, ideally what should have been the case. So, this distortion evidence is another objective of our mapping.

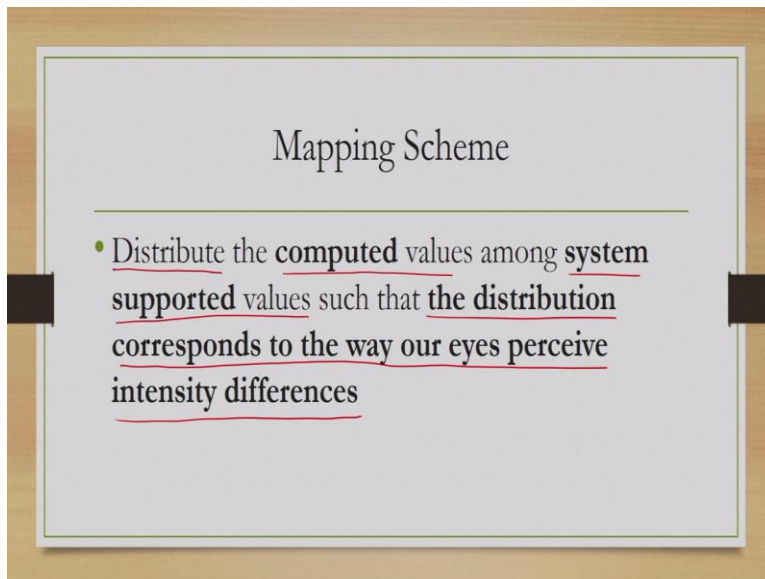
So, we need to map and we need to map in a way such that this distortion is not there. How we can achieve this objective? Let us try to understand the scheme through which we can achieve this objective.

(Refer Slide Time: 13:58)



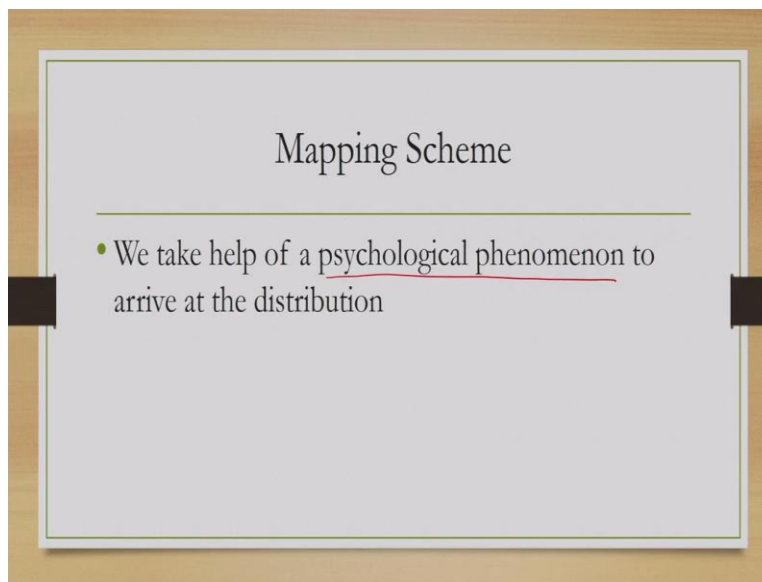
So, that is the Mapping scheme.

(Refer Slide Time: 14:01)



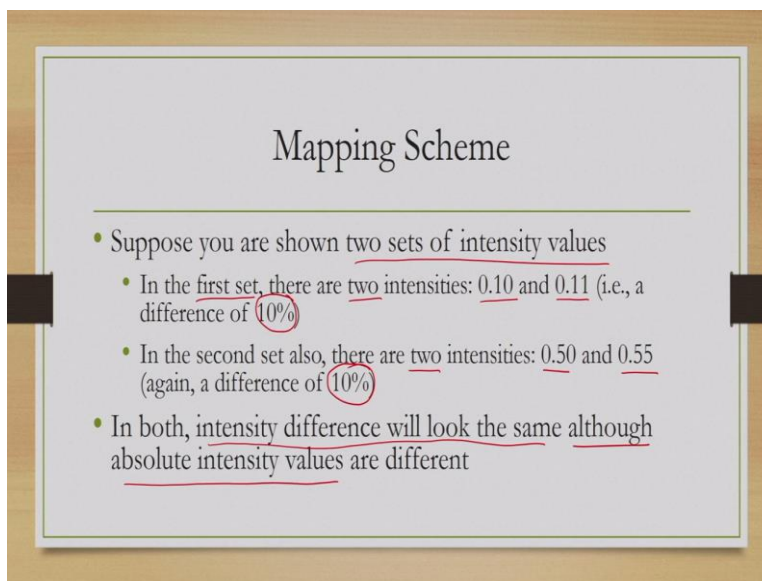
The core idea behind the scheme is that we need to distribute the computed values among the system supported values such that the distribution corresponds to the way our eyes perceive intensity difference. So, this is a slightly complex idea. Let us try to understand this in terms of some example.

(Refer Slide Time: 14:37)



Now, this core idea actually relies on our psychological behavior. How we perceive intensity differences

(Refer Slide Time: 14:58)



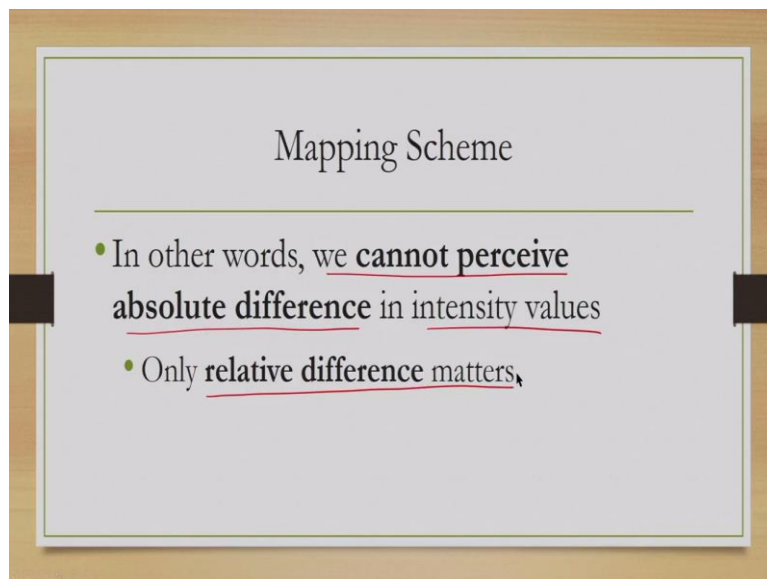
Let us take one example. Suppose, there are two sets of intensity values. In the first set, there are two intensities 0.1 and 0.11. So, the difference between the two intensities is 10 percent. In the second set also, there are two intensities 0.5 and 0.55, again here the difference is 10 percent. But, due to our psychological behavior, we will not be able to perceive the absolute difference

between the intensity values, the difference will look the same, although absolute values are different.

So, in first case we have two absolute values, although the relative difference between them is 10 percent. And in the second set, we have two absolute values which are different than the first set, but the relative difference is same 10 percent.

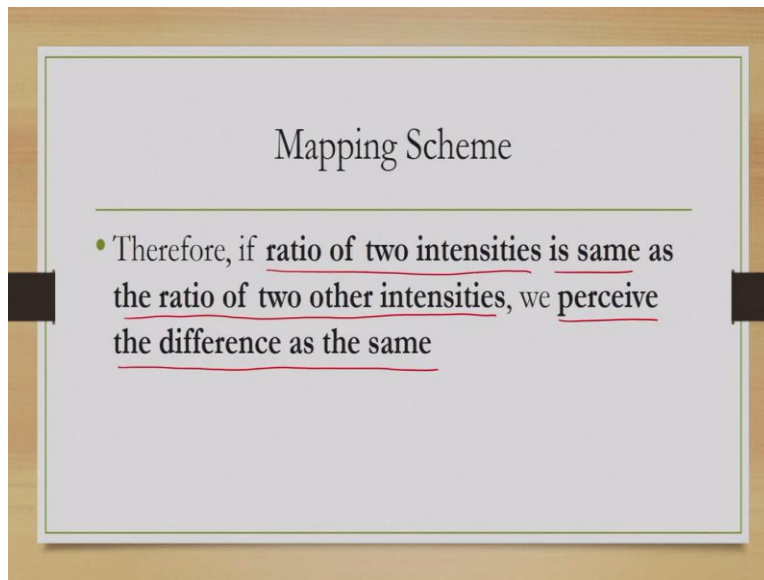
If we are asked to look at those two sets of values, we will not be able to perceive the difference between those values because of our psychological behavior, that we do not perceive the absolute differences, instead, we perceive the relative differences. If the relative differences are same, then we will not perceive any difference if in spite of absolute differences being there.

(Refer Slide Time: 16:45)



So, that is one crucial behavioral trait of us, we cannot perceive the absolute difference in intensity values only relative difference matters. Now, if that is the case, then we can utilize this knowledge to distribute the intensity values among the device supported intensity values. How we can do that?

(Refer Slide Time: 17:17)

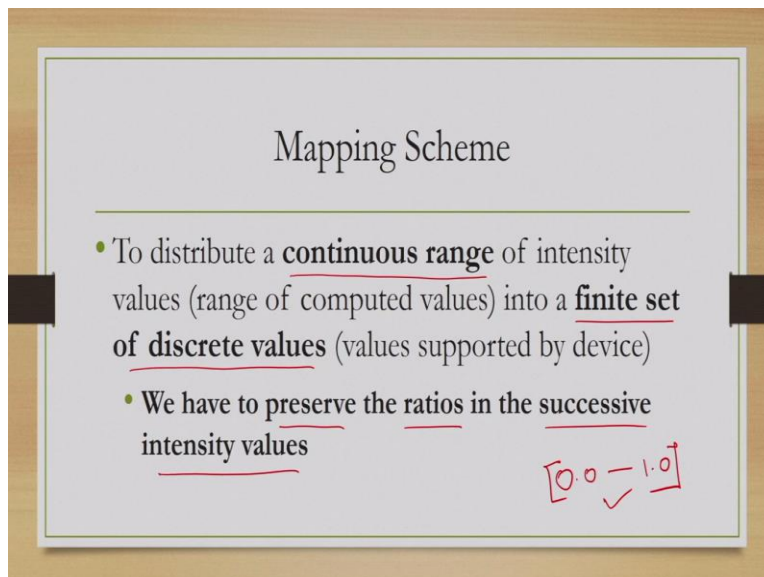


Mapping Scheme

- Therefore, if ratio of two intensities is same as the ratio of two other intensities, we perceive the difference as the same

It follows from our behavioral trait, that if ratio of two intensities is the same as the ratio of two other intensities, then we perceive the difference as the same. This is an implication of the psychological behavior that we just described. And using this implication, we can distribute the intensities, let us see how.

(Refer Slide Time: 17:56)



Mapping Scheme

- To distribute a continuous range of intensity values (range of computed values) into a finite set of discrete values (values supported by device)
  - We have to preserve the ratios in the successive intensity values

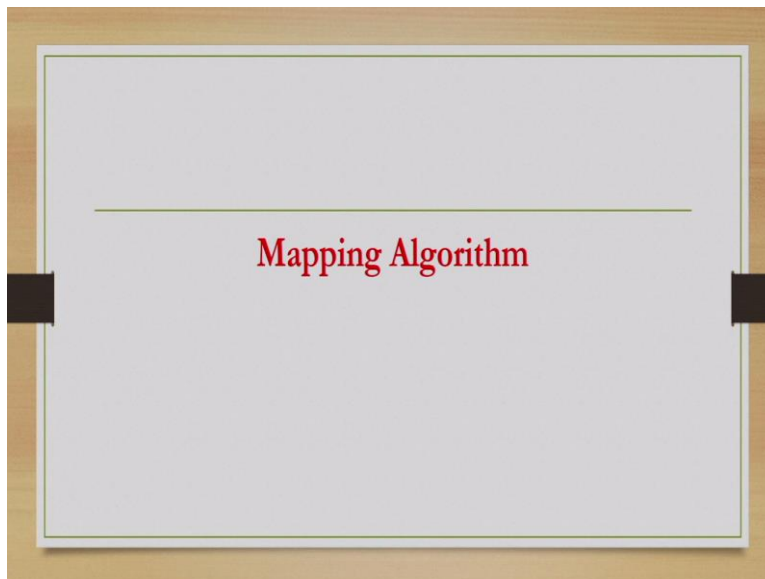
$[0.0 - 1.0]$

Recall that we are given a continuous range of values between 0.0 and 1.0. So, this is our range of computed intensity values computed using Lighting or Shading model. On the other hand, the device supports a set of discrete values, because the frame buffer is designed in that way. And

we are supposed to map this continuous range to that set of discrete values. This continuous range needs to be distributed into the finite set of discrete values.

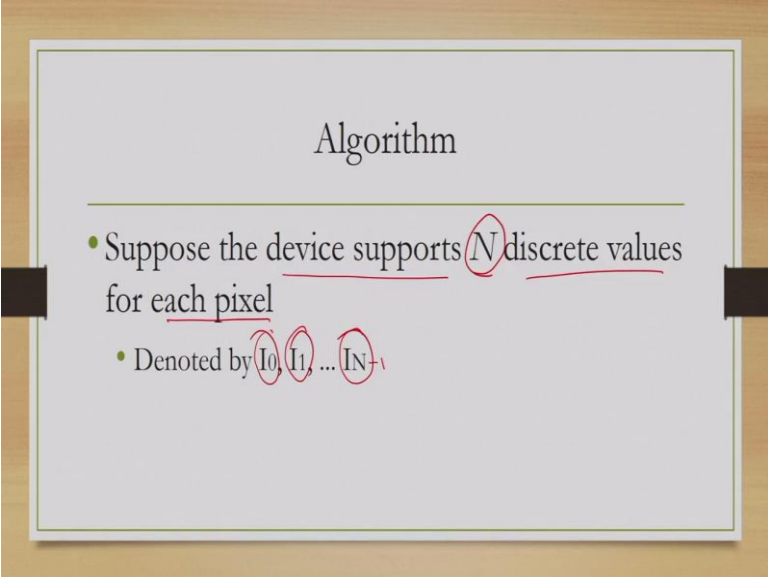
And we can do that without distorting the image by preserving the ratios in the successive intensity values, if we preserve the ratio in the successive intensity values, then even if we are approximating a computed intensity to a device supported intensity, the resulting image will not appear to be distorted and this comes from the psychological trait that we have just discussed. That is our eyes are not designed to perceive absolute differences in intensities; instead, only relative differences matter.

(Refer Slide Time: 19:35)



So, based on this reasoning, we can come up with a mapping algorithm, a step by step process to map a computed intensity to one of the device supported intensities.

(Refer Slide Time: 19:52)

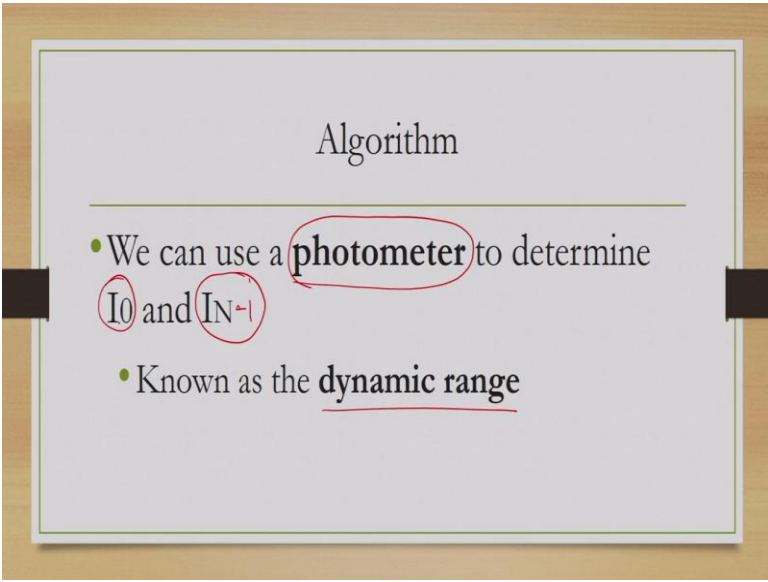


Algorithm

- Suppose the device supports  $N$  discrete values for each pixel
  - Denoted by  $I_0, I_1, \dots, I_{N-1}$

Let us assume that the device supports  $N$  discrete values for each pixel and let us denote these values by  $I_0, I_1$ , up to  $I_N$ . So, there are it should be  $N-1$ . So, denoted by  $I_0, I_1$ , up to  $I_{N-1}$ , there are  $N$  discrete values.

(Refer Slide Time: 20:23)



Algorithm

- We can use a **photometer** to determine  $I_0$  and  $I_{N-1}$ 
  - Known as the dynamic range

Now, we can use a particular device called a Photometer to determine the boundary values that is  $I_0$  and  $I_{N-1}$ . Now, it means that we know the range of intensities supported by that particular system; this is called the Dynamic range, which is bounded by  $I_0$  and  $I_{N-1}$ .

(Refer Slide Time: 21:12)

Algorithm

---

- The highest value in the range is usually taken as 1.0
- Thus, intensities range between  $I_0$  to 1.0

$[I_0, 1.0]$

Now, the highest value that is  $I_{N-1}$  is usually taken to be 1.0 that is a convention used. So, the intensities range between  $I_0$  and 1 this is the range  $[I_0, 1]$ . This is the dynamic range. And  $I_0$  value we can obtain by using the particular device called Photometer.

(Refer Slide Time: 21:44)

Algorithm

---

- To preserve ratio between successive intensities, following must hold

$$\frac{I_1}{I_0} = \frac{I_2}{I_1} = \dots = \frac{I_{N-1}}{I_{N-2}} = r$$

- $r$  is common ratio

Now, we will apply the knowledge that we have just discussed that is to preserve the ratio between successive intensities, we must ensure the following that is a  $I_1/I_0 = I_2/I_1 \dots I_N/I_{N-2} = a$  common ratio  $r$ . So, the ratio of the consecutive intensity values supported by the device should be the same.



(Refer Slide Time: 22:26)

Algorithm

---

- Thus, we have

$$I_1 = rI_0$$
$$I_2 = rI_1 = r(rI_0) = r^2I_0$$
$$I_3 = rI_2 = r(r^2I_0) = r^3I_0$$

...

In other words, we can express all intermediate values in terms of the lowest value. So,  $I_1$  we can represent by this expression  $rI_0$ ,  $I_2$  similarly, we can express by the expression  $r^2I_0$ ,  $I_3$  to be  $r^3I_0$ , and so on.

(Refer Slide Time: 22:52)

Algorithm

---

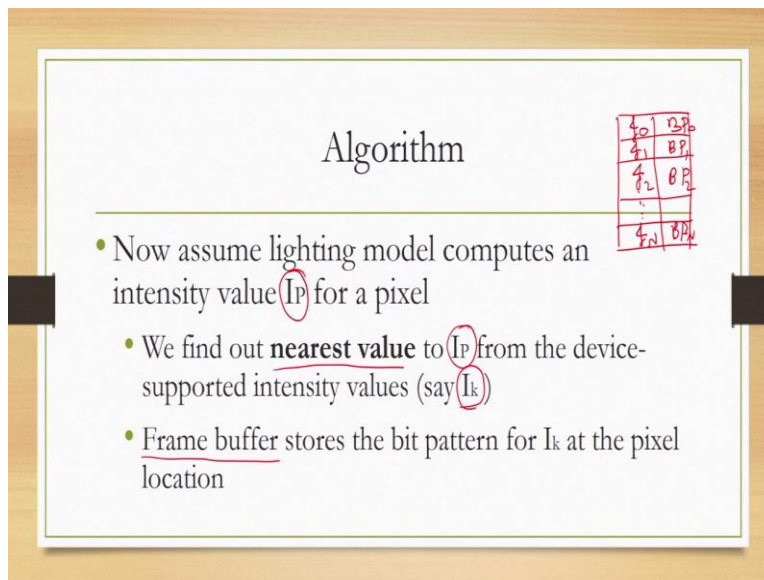
- In general,  $I_k = r^k I_0$  for  $k > 0$
- Thus, we can say,  $I_N = r^N I_0$
- Since we have already determined  $I_0$  and  $I_N = 1.0$ , we can determine  $r$   
 $1.0 = r^N I_0$
- Using  $r$ , we can obtain  $N$  intensity values (and assign bit patterns to those)  
 $N+1$

So, in general, we can say that this equation holds that is  $I_k$  is equal to  $r^k$  and  $I_0$  where  $I_0$  is the minimum intensity for  $k > 0$ . Going along this line we can say  $I_N$  is equals to  $r^N I_0$ . So, this equation holds for any intensity value supported by the device. Now, here you can notice that the

total number of intensity values supported by the device is represented by  $N+1$  and  $I_N$  is the maximum intensity value,  $I_0$  is the minimum intensity value.

So, then what we need to do as we already discussed, we have already determined the minimum value, and we assume the maximum value to be 1. Minimum value we determined using a photometer and we assuming maximum value to be 1. Then, using this equation, we can determine the value of  $r$  by solving the equation  $1 = r^N I_0$  where we know the value of  $I_0$  and we know the value of  $N$  from the total number of intensity values supported by the device. Then using this value of  $r$ , which we compute by solving this equation, we can obtain the  $N$  intensity values using this equation for any particular intensity value  $k$ .

(Refer Slide Time: 25:43)



**Algorithm**

$I_0$	BP
$I_1$	BP
$I_2$	BP
...	
$I_N$	BP

- Now assume lighting model computes an intensity value  $I_p$  for a pixel
  - We find out nearest value to  $I_p$  from the device-supported intensity values (say  $I_k$ )
  - Frame buffer stores the bit pattern for  $I_k$  at the pixel location

Now, let us try to understand what to do next, what should be our next step. So, in the previous step we computed the value of  $r$  knowing the minimum value, maximum value and the total number of intensity values supported by the device. Then based on that we can compute any  $I_k$ . Now, suppose, using a Lighting model, we computed an intensity value for a pixel to be  $I_p$ . So, we are denoting this intensity value by  $I_p$ .

Now, we will maintain a table, in the table, what we will do, we will maintain the intensity values supported by the device, which we compute using the earlier equation. So, that is  $I_0$  which we get with photometer  $I_1, I_2$  in this way to  $I_N$ , then once we compute  $I_p$ , we will try to see from

this table which value comes closest to  $I_p$ . That is, we will try to find out the nearest value that is closest to  $I_p$ .

Let us call it  $I_k$ . Now, for that value, we already have a bit pattern stored here in this table. Let us call it bit pattern 0, bit pattern 1, bit pattern 2, this way bit pattern N for the N+1 intensity values. So, for the  $k^{\text{th}}$  intensity value in the table  $I_k$  we know the corresponding bit pattern. So, then we take that bit pattern and store it in the frame buffer. So, that is how we map value computed using a Lighting model to a bit pattern that represents a value supported by the device.

(Refer Slide Time: 28:23)

### Algorithm Steps

---

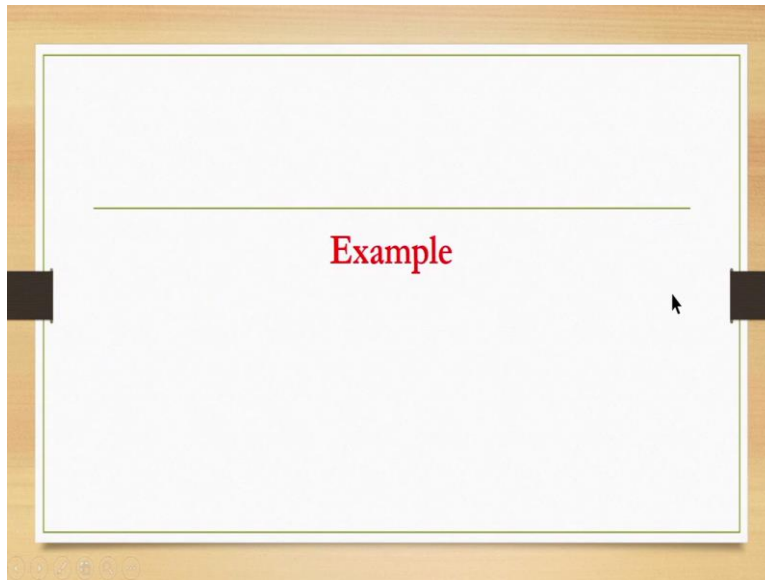
- Determine  $N$  and  $I_0$  ( $I_N = 1.0$ )
- Compute  $r$  (from relation  $I_N = r^N I_0$ )
- Determine device intensities and bit patterns
- Map computed intensity to nearest device intensity

$I_0$  —  $BP_0$   
 $I_1 = r I_0$  —  $BP_1$   
 $I_2 = r^2 I_0$  —  $BP_2$   
 $\vdots$   
 $I_N$  —  $BP_N$

So, then in summary what we do, we determine the value of N and the minimum value  $I_0$  using photometer and assume  $I_N$  to be 1.0 that is the maximum value to be 1.0. Then using the equation  $I_N = r^N I_0$ , we solve for the value r. Then using the value of r we calculate the device supported intensity values. So, we know  $I_0$ , then we calculate  $I_1$  to be  $r I_0$ ,  $I_2$  to be  $r^2 I_0$ , and so on.

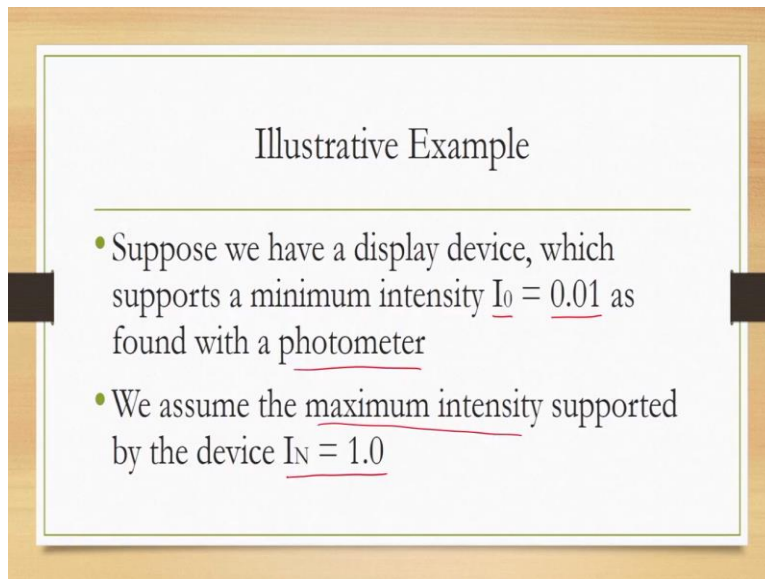
And for each of these computed values, we keep a bit pattern. So, this is our table upto bit pattern for the maximum value. Then we compute for a pixel the intensity value using a Lighting model, map it to the nearest device supported intensity value by looking at the table and then we use the corresponding bit pattern to represent that computed intensity value and finally, we stored that value in the frame buffer. That is how we map computed intensity value to a bit pattern and store it in the frame buffer location.

(Refer Slide Time: 30:18)



Let us try to understand this whole process using one example.

(Refer Slide Time: 30:25)



Suppose, we have a display device, which supports a minimum intensity  $I_0$  as 0.01 and this value of course, as we mentioned earlier we found out with the photometer device. As usual, we are assuming that the maximum intensity value supported by the device to be  $I_N$  equal to 1.0.

(Refer Slide Time: 30:58)

Illustrative Example

---

- The device supports 8 bits for each pixel location
- Then number of intensity values ( $M=N+1$ ) supported by the device for each pixel is  $2^8 = 256$  (i.e.,  $N = 255$ )

Let us assume that the device supports 8 bits for each pixel location. In other words, it has 8 bits to represent the color of a pixel then the total number of intensity values, which we can denote by  $M$  to be  $N+1$  as discussed earlier supported by the device for each pixel is  $2^8$  or 256. So,  $M=256$  that means  $N$  is 255. So, from  $I_0$  to  $I_{255}$  are the intensity values that will be supported by the device.

(Refer Slide Time: 31:59)

Illustrative Example

---

- Thus, the intensities values are  $I_0, I_1, \dots, I_{255}$
- Then,  $I_0 = r^{255} \cdot 0.01$  ✓  
 $I_N = r^N \cdot I_0$

So, these intensity values we can denote with these notations  $I_0, I_1, I_2$  up to  $I_{255}$ . Now, we can set up this equation based on the relationship that is  $I_N = r^N I_0$ . Now, here we are replacing the values

$I_N$ ,  $I_0$  and  $N$  to get this equation and we solve this equation to get the value of  $r$ . So, if you solve it, you will get the value of  $r$ .

(Refer Slide Time: 32:38)

Illustrative Example

---

- Solving, we get  $r = \underline{1.0182}$
- Thus,
  - $I_1 = 1.0182 * 0.01 = \underline{0.0102}$
  - $I_2 = r * I_1 = 1.0182 * 0.0102 = \underline{0.0104}$
  - And so on

So, solving this we get  $r = 1.0182$  and using this value, we get other intensity values in this way, so,  $I_1$  will be  $rI_0$  that is 0.0102.  $I_2$  will be  $r^2I_0$ , that is 0.0104, and so on. And we create a table of these values.

(Refer Slide Time: 33:15)

Illustrative Example

---

- Next, we map to bit patterns
- One possible mapping

Intensity	Bit pattern
$I_0 = 0.0100$	<u>00000000</u>
$I_1 = 0.0102$	<u>00000001</u>
$I_2 = 0.0104$	<u>00000010</u>
...	...
$I_{255} = 1.0$	<u>11111111</u>

In this table also we assigned bit patterns. So,  $I_0$  we assigned 000,  $I_1$  we assign this bit pattern  $I_2$  we assign this bit pattern and so on up to this bit pattern for the last value. This is of course one possible mapping. Now, assignment of bit pattern can be arbitrary, it really does not matter because it has nothing to do with the preservation of ratio. But the actual calculation of these intensity values is what matters. This calculation is done based on the principle of preserving the ratios of successive intensity values. So, that the resulting image is not distorted.

(Refer Slide Time: 34:09)

34:31 / 42:22

### Illustrative Example

- Suppose we computed intensity at a pixel location as 0.01039 (with the lighting model)

Intensity	Bit pattern
$I_0 = 0.0100$	00000000
$I_1 = 0.0102$	00000001
$I_2 = 0.0104$	00000010
...	...
$I_{255} = 1.0$	11111111

Now, let us assume that we have computed some intensity values using the Lighting model at a pixel location and that value is 0.1039. So, this is our table, and we computed this value.



(Refer Slide Time: 34:33)

Illustrative Example

---

- We try to find nearest intensity pixel supports
  - In this case, it is 0.0104 ( $I_2$ )
  - Hence, pixel intensity represented by the bit pattern 0000010

Intensity	Bit pattern
$I_0 = 0.0100$	00000000
$I_1 = 0.0102$	00000001
$I_2 = 0.0104$	<u>00000010</u>
...	...
$I_{255} = 1.0$	11111111

So, as per the algorithm what we should do, we try to find out the nearest intensity value that the pixel support. So, in this case, that is  $I_2$  or 0.104, and the bit pattern corresponding to  $I_2$  is this one. So, we store this bit pattern at the corresponding frame buffer location.

(Refer Slide Time: 34:58)

Illustrative Example

---

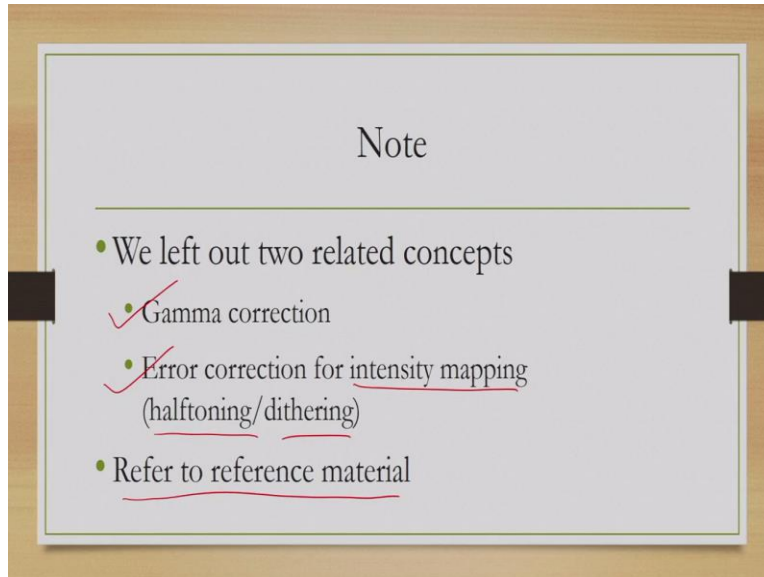
- Note - final intensity different from computed intensity

So, here you may note that the final intensity value that we represented and stored in the frame buffer is different from the actual value that is computed using the Light model because of the mapping. So, it means that there is some error, always there will be some error. Although with the preservation of the ratio of successive intensities, we can elevate the problem of visual



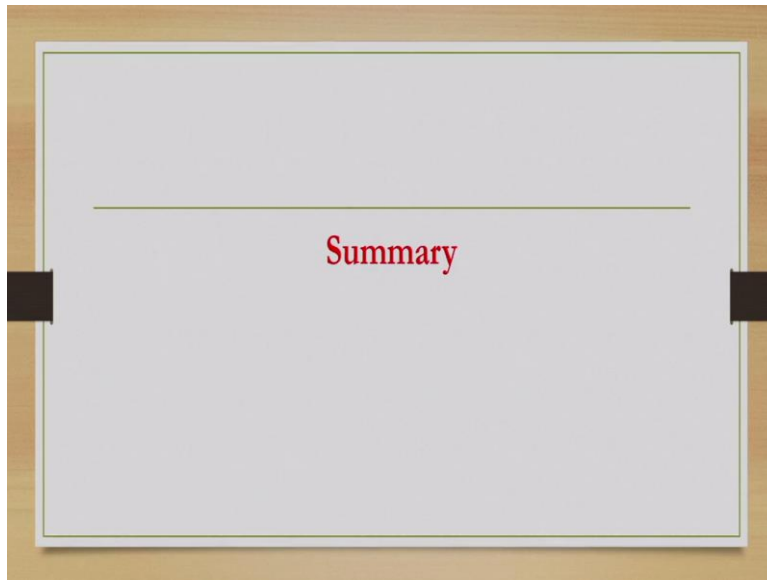
distortion in the resulting image. Still, there are ways to improve this selection of appropriate intensity representing a computed intensity.

(Refer Slide Time: 35:57)



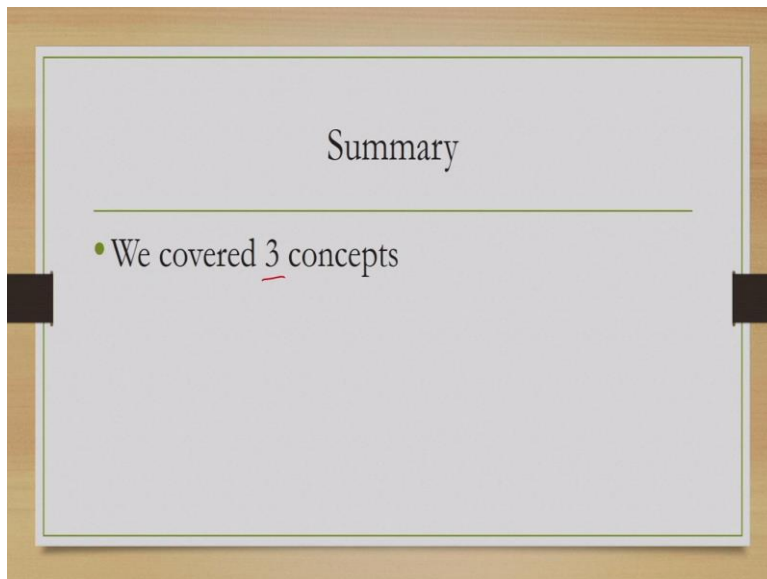
And there are some techniques to do that, at different level, one is Gamma correction, other one is Error correction for intensity mapping through halftoning or dithering methods. However, we will not go into the details of these methods. The basic idea is that using these methods, we can actually reduce the effect that arises due to the introduction of mapping errors, the difference between computed intensity, and the intensity that we represent and store in the frame buffer. If you are interested, you may refer to the material that will be mentioned at the end of this lecture.

(Refer Slide Time: 36:48)



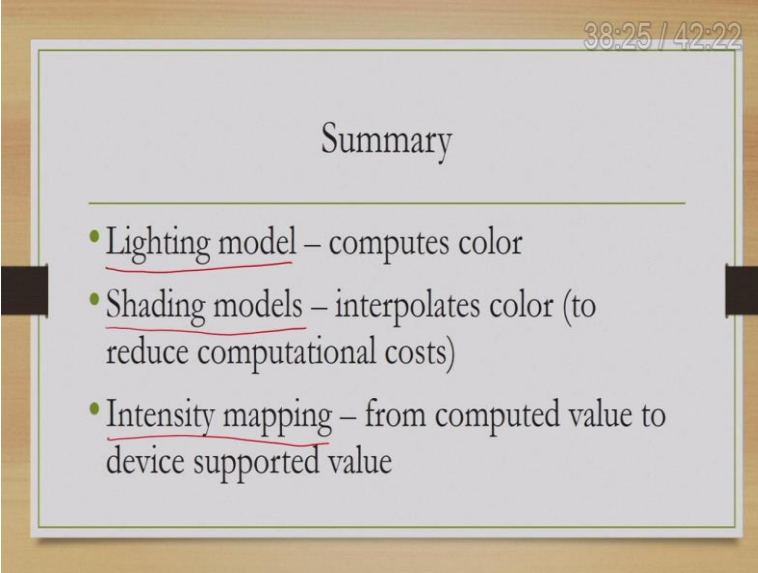
So, in summary, what we can say is that.

(Refer Slide Time: 36:53)



In stage three, there are three broad concepts that we have covered, what are these concepts.

(Refer Slide Time: 37:09)



38:25 / 42:22

### Summary

- Lighting model – computes color
- Shading models – interpolates color (to reduce computational costs)
- Intensity mapping – from computed value to device supported value

The first is Lighting model. So, this is the basic method that we follow to simulate the optical properties and behavior which gives us the sensation of color. Now, lighting model is complex. So, in order to avoid complexities, we take recourse to Shading models. This is the second concept that we have learned. Shading models is essentially a way to reduce computation, while assigning colors to surface points, it makes use of lighting models, but in a very limited way and uses interpolation, which are less computation intensive to assign colors to surface points.

Then the third concept that we have discussed is Intensity Mapping. So, with Lighting or Shading model we compute color as a real number within a range of 0 to 1. So, any value can be computed. However, a computer does not support any value, it is discrete in nature. So, essentially discrete values are supported a subset of values of all possible values are supported.

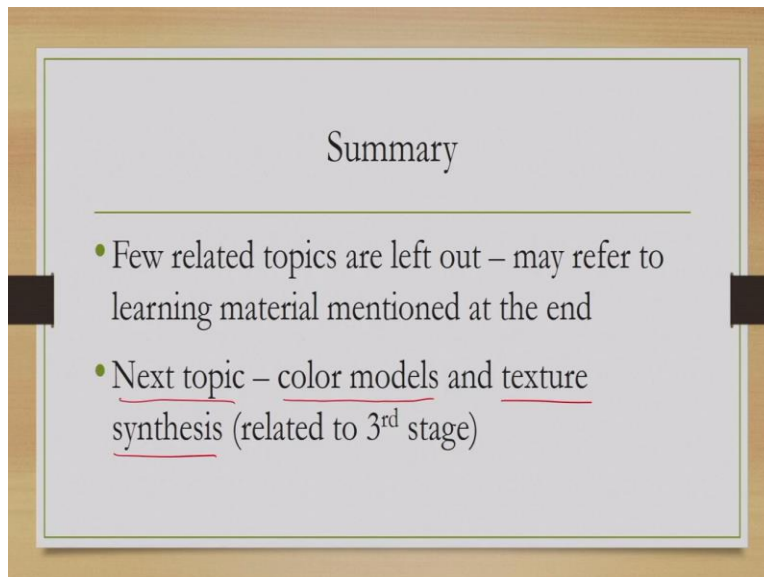
For example, if we have 8 bit frame buffer that means each pixel location is represented by 8 bits we can support at most 256 intensity values for each pixel. A pixel color can be any one of these 256 values, whereas, we are computing color as any value between 0 to 1. So, we need to map it, this mapping is complex and it introduces some amount of error. This error may result in distortion.

However, to avoid distortion, we make use of one psychological behavioral aspect of our visual perception. That is, we distribute the computed or potential intensities among the device supported intensities in a way such that the ratio of the consecutive intensities remains the same.

If we do that, then this perceived distortion of the image may be avoided. However, in spite of that, we introduce some error which may affect the quality of the image.

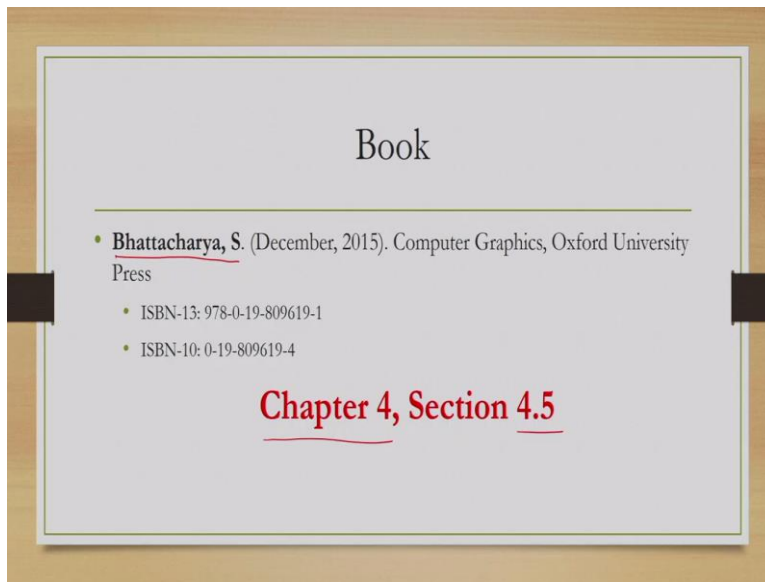
Because whatever color we are computing, we are not actually assigning exactly the same color to the final image, instead, we are mapping it to a nearest color. So, in turn it may affect the overall quality. To reduce that, few techniques are used like Gamma correction or Error propagation. And these techniques you can go through on your own in the reference material that we will mention at the end. In this lecture, we will not go into the details of those techniques, as those are not relevant for our discussion.

(Refer Slide Time: 40:44)



In the next lecture, what we will do is we will discuss another important aspect of the third stage that is Color model. Along with that, we will also learn about Texture synthesis, both are part of the third state that is coloring. So, so far we have learned three concepts and two more concepts we will learn in the subsequent lectures.

(Refer Slide Time: 40:44)



Whatever we have discussed today can be found in this book. You may refer to Chapter 4, Section 4.5, to learn about the topics, and also you may find more details on the topics that we mentioned, but we did not discuss in details, namely the Error propagation techniques and Gamma correction techniques. That is all for today. Thank you and goodbye.