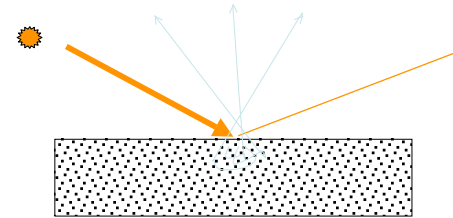


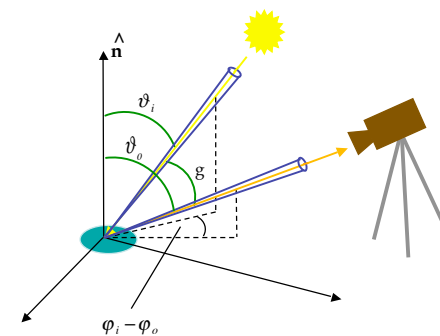
## Light interacting with the world

- The light captured by camera carries information about what is in the world **because** what is in the world interacts with it differently depending on what it is.
- Many effects when light strikes a surface. It could be:
  - absorbed
  - transmitted
  - reflected
  - scattered (in a variety of directions!)



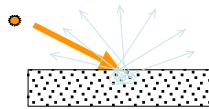
## Bidirectional Reflectance Distribution Function (BRDF)

- The BRDF is a technical way of specifying how light from sources interacts with the matter in the world
- Understanding images requires understanding that this varies as a function of materials. The following “look” different
  - mirrors
  - white styrofoam
  - colored construction paper
  - colored plastic
  - gold
- The BRDF is the **ratio** of what comes out to what came in
- What comes out  $\leftrightarrow$  “radiance”
- What goes in  $\leftrightarrow$  “irradiance”
- Details on the BRDF available as supplementary material



## Lambertian surfaces

- Simple special case of reflectance: ideal diffuse or matte surface--e.g. cotton cloth, matte paper.
- Surface appearance is independent of viewing angle.
- Typically such a surface is the result of lots of scattering---the light "forgets" where it came from, and it could end up going in any random direction.



- What counts is how much light power reaches the surface

## Lambertian surfaces

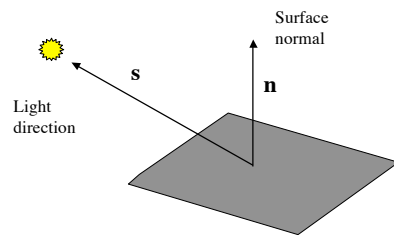
Simple rule for shading--attenuate brightness by

$$\mathbf{n} \cdot \mathbf{s}$$

↑ Surface normal
 ↑ Light source direction

Must know this

## Lambertian Reflection

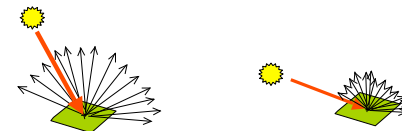


Brightness is proportional to  $\mathbf{n} \cdot \mathbf{s}$

## Lambertian Reflection

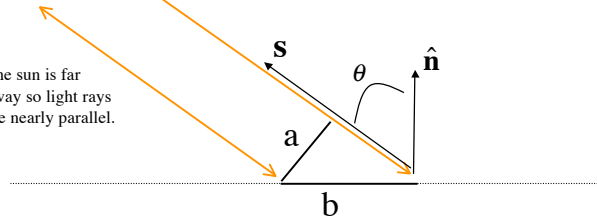
Why is brightness proportional to  $\mathbf{n} \cdot \mathbf{s}$  ?

Intuitive argument: The surface scatters light in all directions equally, but as the angle of the light becomes oblique, the amount of light per unit area received is reduced (foreshortened) by a factor of the cosine of the angle.





The sun is far away so light rays are nearly parallel.



The same light is spread over  $a$ , giving intensity,  $i_a$ , as is spread over  $b$ , giving intensity,  $i_b$ . This means that:

$$a \cdot i_a = b \cdot i_b$$

or, because  $a$  is the length of the perpendicular,

$$i_b = i_a \left( \frac{a}{b} \right) = i_a \cos(\theta)$$

## Lambertian Reflection

Most the world is not Lambertian

Lambertian assumption failures

Rough surfaces--important example--the moon is not Lambertian

Dielectrics (plastics, many paints)

Metallic surfaces

Skin

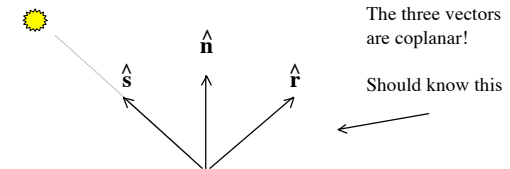
Mirrors

## Ideal Mirrors

The opposite extreme case from Lambertian is a mirror.

Instead of going everyway equally, the reflected light goes exactly one way.

## Reflection Direction



$$\hat{s} + \hat{r} = k\hat{n}$$

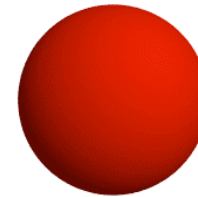
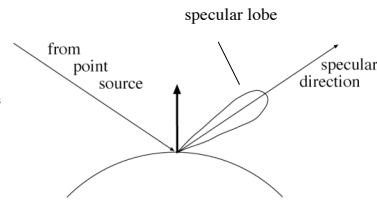
$$\hat{n} \cdot \hat{s} + \hat{n} \cdot \hat{r} = k \Rightarrow k = 2\hat{n} \cdot \hat{s}$$

$$\text{So } \hat{r} = 2(\hat{n} \cdot \hat{s})\hat{n} - \hat{s}$$

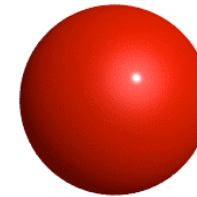
Optional--Just in case you need it for something!

## Specular surfaces

- Another important class of surfaces is specular (somewhat mirror-like).
  - specular surfaces reflect a significant amount of energy in the specular (mirror) direction
  - a significant amount may also be reflected in a direction roughly in the mirror direction (specular lobe)
  - typically there is a diffuse component as well



Diffuse Lighting



Plus Specular Highlight

from

<http://www.geocities.com/SiliconValley/Horizon/6933/shading.html>

## Standard nearby point source model (Lambertian)

If the point source is close, moving around in the scene changes the distance to the source.

$$\rho_d(\mathbf{X}) \left( \frac{\hat{\mathbf{n}}(\mathbf{X}) \cdot \mathbf{s}(\mathbf{X})}{|\mathbf{X} - \mathbf{X}_s|^2} \right)$$

$\mathbf{X}$  is the location in the world

$\mathbf{X}_s$  is the location of the source

$S$  is the strength of the source

$$\mathbf{s}(\mathbf{X}) = s(S, \mathbf{X}_s, \mathbf{X}) = S \hat{\mathbf{s}}(\mathbf{X}_s, \mathbf{X}) = \frac{\mathbf{X}_s - \mathbf{X}}{|\mathbf{X}_s - \mathbf{X}|}$$

$\rho_d(\mathbf{X})$  is the reflectance ratio at  $\mathbf{X}$ .

$$\rho_d(\mathbf{X}) \left( \frac{\hat{\mathbf{n}}(\mathbf{X}) \cdot \mathbf{s}(\mathbf{X})}{r(\mathbf{X})^2} \right)$$

## Standard distant point source model

- If the source is far away, this formula reduces to the same as the Lambertian formula from before:

then  $|\mathbf{X} - \mathbf{X}_s| \equiv R$  (distance does not change much)

and  $\mathbf{s}(\mathbf{X}) \equiv \mathbf{S}$  (nor does direction)

if we let  $\mathbf{S}_d = \left( \frac{\mathbf{S}}{R^2} \right)$

we get  $\rho_d(\mathbf{X}) (\hat{\mathbf{n}}(\mathbf{X}) \cdot \mathbf{S}_d)$  (as before)



## More General Illumination

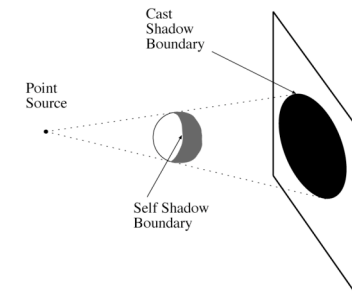
The effect of light is generally linear (**know this**)!

We can break the effect of multiple sources into a sum, with one component for each source of light.

If we want to model an extended source, we add up (i.e., integrate) the contributions to small (e.g. point) bits, using the point source model.

## Shadows cast by a point source

- A point that can't see the source is in its shadow
- For point sources, the geometry is simple
- For extended sources, we have an **umbra** (points seeing no light), and a **penumbra** (seeing some parts of the light but not all)



## The Shadow Problem



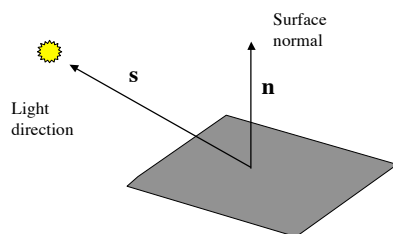
## The Shadow Problem



## Shape from shading



- Suppose that we know the direction of a point source (there are ways to guess)
- Suppose Lambertian reflectance
- We know that image pixel brightness is proportional to  $\mathbf{n} \cdot \mathbf{s}$
- Can we figure out shape from brightness?



## Shape from shading



- Problems
  - Can we find the normals at every point.
  - Do the normals give us shape? How?

## Shape from shading



- Can we find the normals at every point?
  - Under-constrained! (Only have one piece of data per pixel, but we need 2 or 3 (if we need to estimate albedo as well)).
  - Can impose regularization (smoothness) and consider boundary conditions
- Do normals give us shape?
  - Normals are not shape, but they can be related to the partial derivatives of the shape as a function--the surface is given by  $(x, y, f(x, y))$
  - The partial derivatives must satisfy integrability constraints--random normals do not come from a continuous surface!

## Photometric Stereo

- Shape from shading is hard! Consider an easier problem.
- Suppose that we have a number of known point sources, and we have successive pictures taken with each one used in turn.
- Let  $\mathbf{g}(x, y)$  be the surface normal times the albedo (for the point in the world corresponding to image point  $(x, y)$ ).
- Let  $\mathbf{V}_i$  be the light source direction,  $i$ , times a scalar embodying the light source magnitude and camera sensitivity
- Let  $I_i(x, y)$  be image intensity

Then 
$$I_i(x, y) = \mathbf{V}_i \cdot \mathbf{g}(x, y) \quad (\text{Lambert's law})$$

### Photometric Stereo

$$I_i(x, y) = \mathbf{V}_i \bullet \mathbf{g}(x, y) \quad (\text{Lambert's law})$$

So how to solve for the surface?

Simpler---how to solve for  $\mathbf{g}(x, y)$ ?

How many lights do we need  
(assuming that albedo is not known) ?

### Photometric Stereo

$$I_i(x, y) = \mathbf{V}_i \bullet \mathbf{g}(x, y) \quad (\text{Lambert's law})$$

Hopefully this looks like the  $i$ 'th row of matrix,  $\mathbf{V}$ , multiplied by a vector,  $\mathbf{g}(x, y)$ .

### Photometric Stereo

Thus combining the conditions given by each light,  $i$ , we get

$$\mathbf{i} = \mathbf{V}\mathbf{g}$$

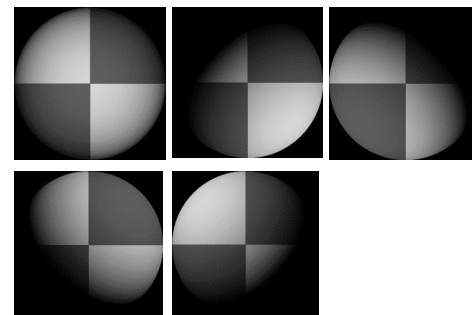
Where the  $i^{\text{th}}$  element of  $\mathbf{i}$  is  $I_i(x, y)$  and the  $i^{\text{th}}$  row of  $\mathbf{V}$  is  $\mathbf{V}_i$

Since  $\mathbf{g}$  has three elements, we need at least 3 lights.

If the number of lights is more than 3, then use least squares!

You should understand the construction of this problem.

### Example figures



## Dealing with shadows

Each point is in K images (one for each light)

If  $I_i(x,y)$  is in shadow, then ignore it

Can simplify this in a program with a shadow matrix

## Dealing with shadows

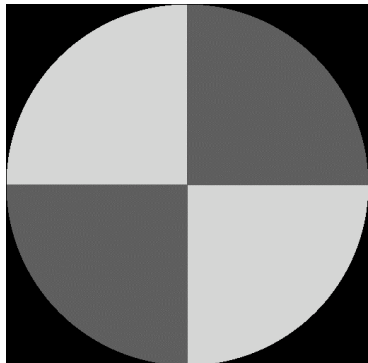
$$\begin{pmatrix} I_1^2(x,y) \\ I_2^2(x,y) \\ \vdots \\ I_n^2(x,y) \end{pmatrix} = \begin{pmatrix} I_1(x,y) & 0 & \dots & 0 \\ 0 & I_2(x,y) & \dots & \dots \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \dots & 0 & I_n(x,y) \end{pmatrix} \begin{pmatrix} \mathbf{V}_1^T \\ \mathbf{V}_2^T \\ \vdots \\ \mathbf{V}_n^T \end{pmatrix} \mathbf{g}(x,y)$$

$\nwarrow$  known image intensity       $\updownarrow$  shadow  $\Rightarrow$  0 otherwise 1       $\updownarrow$  known light vectors       $\nearrow$  Unknown

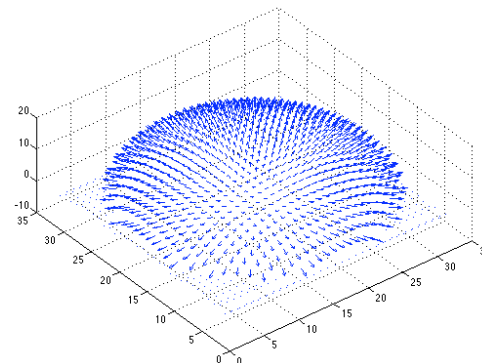
If the intensity is small, then set it to zero, and set the entry in the shadow matrix to zero.

(Fancy way to zero out the corresponding rows)

## Recovered reflectance



## Recovered normal field



## From Normals to Shape

From  $\mathbf{g}$  we can get the normal  $\hat{\mathbf{n}} = \frac{\mathbf{g}}{|\mathbf{g}|}$

It is natural to represent surface as a depth map  $(x, y, f(x, y))$

But what is the relationship between that and the normals?

## From Normals to Shape

Given  $(x, y, f(x, y))$ , what is the surface normal direction?

## From Normals to Shape

Given  $(x, y, f(x, y))$ , what is the surface normal direction?

Method one (cross product)

$$\hat{\mathbf{n}} \propto \frac{\partial}{\partial x}(x, y, f(x, y)) \times \frac{\partial}{\partial y}(x, y, f(x, y))$$

(If you are on the surface the partial with respect to  $x$  gives the direction in 3D if you try to go in the  $x$  direction. Ditto for  $y$ . The normal is perpendicular to these two directions).

## From Normals to Shape

Given  $(x, y, f(x, y))$ , what is the surface normal direction?

Method two (level curves)

Given a surface,  $S$ , specified by  $g(x, y, z) = 0$

$\nabla g(x, y, z)$  is normal to  $S$

So, find  $g(x, y, z)$  such that  $g(x, y, z) = 0$  is our surface

## From Normals to Shape

Given  $(x, y, f(x, y))$ , what is the surface normal direction?

Method two (level curves)

$$g(x, y, z) = z - f(x, y)$$

$$\nabla g(x, y, z) = ?$$

## From Normals to Shape

Either way,  $\hat{\mathbf{n}} \propto (-f_x, -f_y, 1)$

It should be clear that  $f_x = -\frac{n_x}{n_z}$  and  $f_y = -\frac{n_y}{n_z}$

(In general,  $\mathbf{n}$  will embody the albedo, so we must be prepared for an arbitrary scale factor in  $\mathbf{n}$ ---most easily dealt with using the above ratio if we want  $f_x$  and  $f_y$ ).

## From Normals to Shape

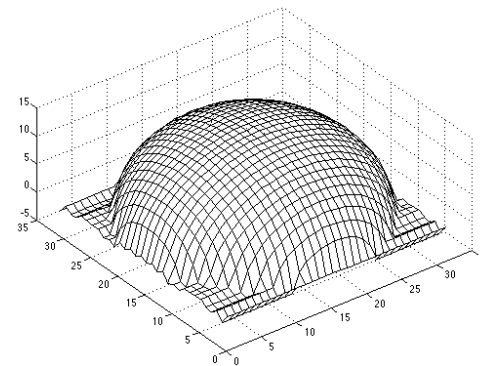
So, if have the normals, we can estimate the derivatives of  $f(x, y)$

Minor point for those who have vector calculus: If we assume that  $f_x$  and  $f_y$  are the derivatives of a differentiable function,  $f(x, y)$  we can further check (or constrain) that  $f_{xy} = f_{yx}$ .

We can recover the surface height at any point by integration along some path. For example, if we declare the origin to be at height  $C$ , and go along the  $x$  axis, then parallel to the  $y$  axis:

$$f(x, y) = \int_0^x f_x(x', 0) dx' + \int_0^y f_y(x, y') dy' + C$$

## Surface recovered by integration



## Color (very briefly)

Color is a sensation

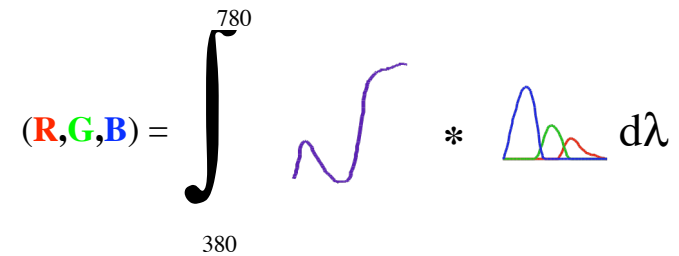
Usually there is light involved, and usually there is a relationship between the world and the colors you see

Your brain has a big effect on the colors you see

We will focus on what colors mean to a camera which is **much simpler**

Color for a camera (R,G,B) is a very limited sampling of spectral light energy (why three values?)

## Recall Image Formation (Spectral)

$$(R, G, B) = \int_{380}^{780} \text{[Spectral Power Distribution]} * \text{[Camera Response Functions]} d\lambda$$


## Recall Discrete Version

Represent the light by a vector,  $\mathbf{L}$

Consider a matrix  $\mathbf{R}$  whose rows are the discretized version of the response functions.

Let  $\mathbf{p}$  be a vector of camera responses (i.e.,  $(R, G, B)^T$ )

Then

$$\mathbf{p} = \mathbf{R} * \mathbf{L}$$

From previous slide

$$\mathbf{p} = \mathbf{R} * \mathbf{L}$$

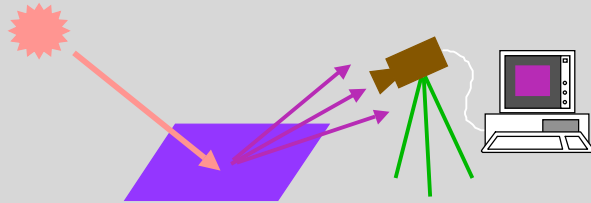
$\mathbf{R}$  is **not** full rank (typical values are 3 by 101 or 3 by 31)

First key observation is that you cannot recover  $\mathbf{L}$  from  $\mathbf{p}$  ( $\mathbf{L}$  is spectra,  $\mathbf{p}$  is RGB)

Second observation---many spectra can have the same RGB.

(This is the essence of color reproduction)

(R,G,B) depends on the light, the surface, and the camera

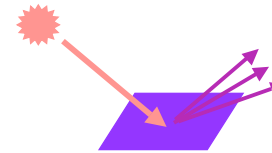


## Spectral reflectance

By definition, the spectral reflectance, satisfies

$$S(\lambda) = \frac{L(\lambda)}{E(\lambda)} \quad \text{where } E(\lambda) \text{ is incoming and } L(\lambda) \text{ is outgoing}$$

So we get  $L(\lambda)$  from before by:  $L(\lambda) = E(\lambda)S(\lambda)$

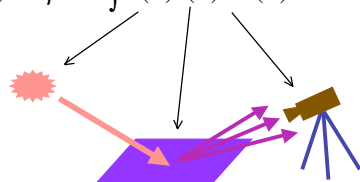


## Spectral reflectance

So we get  $L(\lambda)$  from before by:  $L(\lambda) = E(\lambda)S(\lambda)$

$$\text{Recall } \rho^{(k)} = \int L(\lambda)R^{(k)}(\lambda)d\lambda$$

$$\text{Now, } \rho^{(k)} = \int E(\lambda)S(\lambda)R^{(k)}(\lambda)d\lambda$$



## Naive Color Model

Now consider white light (255, 255, 255)

- This is **relative** to the camera!
- By definition, this is the color of perfect diffuse, uniform, reflector

Suppose that a surface has color ( $R_s, G_s, B_s$ ) under white light

- Naively, this is the “color of the surface”
- (Naïve, because surfaces don’t have color until you turn on the light, and it matters what the color of the light is!)
- The albedo in each channel is  $\rho_R = \frac{R_s}{255} \quad \rho_G = \frac{G_s}{255} \quad \rho_B = \frac{B_s}{255}$



## Naive Color Model (2)

Naive value for the color of the surface under a **different** light,  $(R_L, G_L, B_L)$  is given by:

$$(R, G, B) = (\rho_R R_L, \rho_G G_L, \rho_B B_L)$$

This is naïve because we assume that the part of the light that stimulates one channel, does **not** interact with the albedo of any other channel.

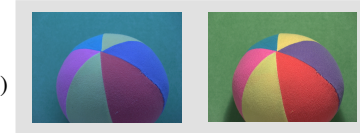
Alternatively, everything about the surface color can be captured in these 3 numbers.

This is the “diagonal model” for illumination change.

## Diagonal Model for Color

(Same scene, but different illuminant)

Light color  
 $(R_{L1}, G_{L1}, B_{L1})$



Light color  
 $(R_{L2}, G_{L2}, B_{L2})$

## Diagonal Model for Color

(Same scene, but different illuminant)

Light color  
 $(R_{L1}, G_{L1}, B_{L1})$



Light color  
 $(R_{L2}, G_{L2}, B_{L2})$

Diagonal model assumes that all the  $(R, G, B)$  in the left image change by the ratio of the lights

$$R_2 = \frac{R_{L2}}{R_{L1}} * R_1 \quad G_2 = \frac{G_{L2}}{G_{L1}} * G_1 \quad B_2 = \frac{B_{L2}}{B_{L1}} * B_1$$

## Diagonal Model for Color

- In matrix form

$$\begin{pmatrix} R_2 \\ G_2 \\ B_2 \end{pmatrix} = \begin{pmatrix} \frac{R_{L2}}{R_{L1}} \\ \frac{G_{L2}}{G_{L1}} \\ \frac{B_{L2}}{B_{L1}} \end{pmatrix} \begin{pmatrix} R_1 \\ G_1 \\ B_1 \end{pmatrix}$$

- Note that this says  $\frac{R_2}{R_{L2}} = \frac{R_1}{R_{L1}}$  (etc, for G, B)

### Diagonal Model for Color

- Note that this says  $\frac{R_2}{R_{L2}} = \frac{R_1}{R_{L1}}$  (etc, for G, B)
- This would mean  $\frac{\int E_2(\lambda)S(\lambda)R^{(k)}(\lambda)d\lambda}{\int E_2(\lambda)R^{(k)}(\lambda)d\lambda} = \frac{\int E_1(\lambda)S(\lambda)R^{(k)}(\lambda)d\lambda}{\int E_1(\lambda)R^{(k)}(\lambda)d\lambda}$  !
- But this is not generally true!

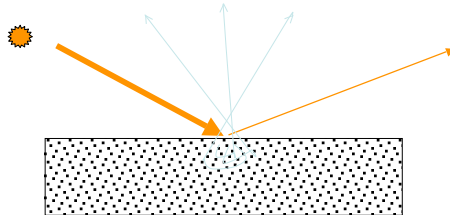
### Diagonal Model for Color

In general,  $\frac{\int E_2(\lambda)S(\lambda)R^{(k)}(\lambda)d\lambda}{\int E_2(\lambda)R^{(k)}(\lambda)d\lambda} \neq \frac{\int E_1(\lambda)S(\lambda)R^{(k)}(\lambda)d\lambda}{\int E_1(\lambda)R^{(k)}(\lambda)d\lambda}$

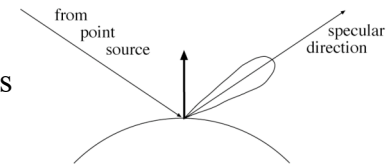
- But expression holds when
  - Surface reflectance is uniform
  - Sensors are delta functions
- Naïve approximation is relatively good when the camera sensors are “sharp” with minimal overlap.

### Color and specularities

- Dielectric surfaces are well approximated by a specular part and a Lambertian body part.



### Specular surfaces

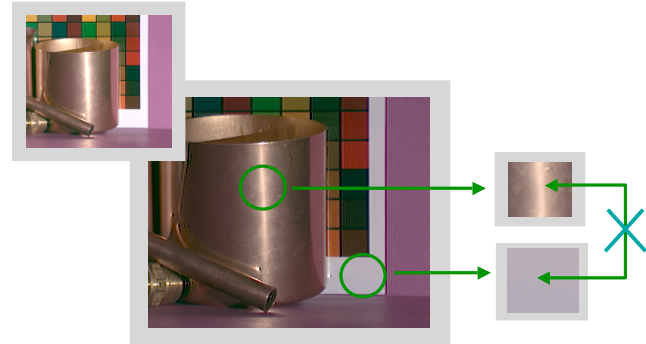


- Important point: The specular part of the reflected light usually carries the color of the **light**
- Technically, this is the case for dielectrics--plastics, paints, glass.
- Important exception is metals (e.g. gold, copper)

### Dielectric Specularities

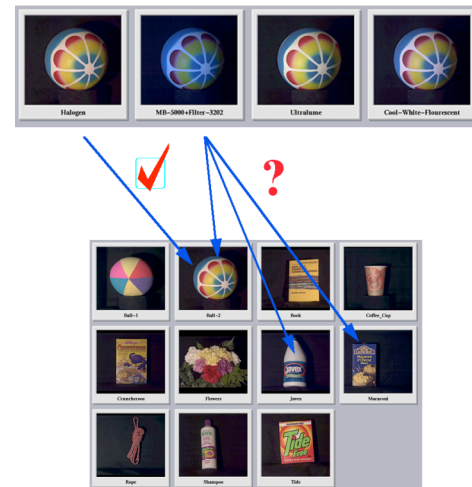


### Metallic Specularities

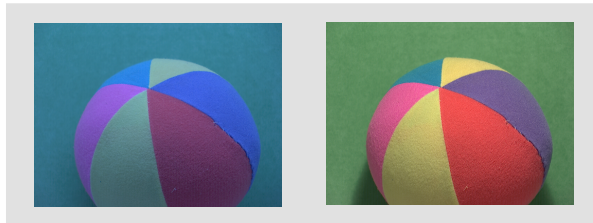


### Color for recognition

- It seems natural to use color (as opposed to grays in a B&W image) to recognize things--why?

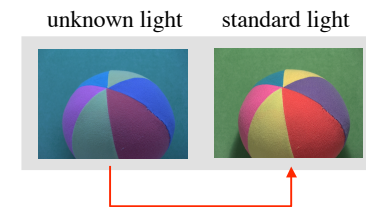


## The Computational Colour Constancy Problem



(Same scene, but different illuminant)

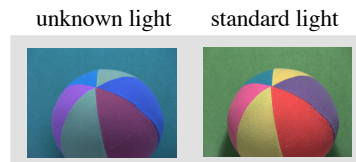
## Color constancy



Color constancy algorithms strive to map image pixels to useful illuminant invariant values. One example is the image as if it was taken under the known standard light.

Often done by estimating the illuminant, followed by color correction (but there are other ways).

## Color Correction



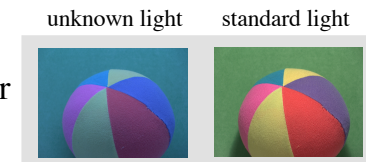
Suppose that the image on the right was how the scene on the left would look under a known, “standard” light.

Under that light a uniform reflective surface (white) is  $(R_w^s, G_w^s, B_w^s)$

So, to correct the image on the left, we can estimate the color of white, under the unknown light. Suppose it is:  $(R_w^u, G_w^u, B_w^u)$

Then we can correct the image using the diagonal matrix from a few slides back.

## Estimating the color of the light



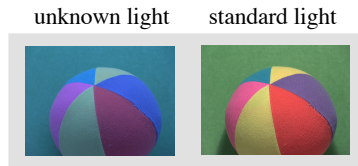
The hard part is to estimate the color of the light (i.e.,  $(R_w^u, G_w^u, B_w^u)$ )

Many interesting algorithms have been developed. Two simple ones:

$$\text{Max RGB: } R_w^u = \max_{\text{pix}}(R) \quad (\text{Similarly for G and B})$$

$$\text{Gray world: } \left(\frac{1}{3}\right)R_w^u = \text{ave}_{\text{pix}}(R) \quad (\text{Similarly for G and B})$$

## Estimating the color of the light



The two simple methods mentioned previously focus on the global statistics of the image.

More formally, one can set up an inference problem focused on estimating the probability that the light is a certain color, **given** the image data.

Another approach is find specularities (recall why this works!).