Graphics & Multimedia

School of Information Technologies
The University of Sydney

Week03 • Semester 2 • 2018

Color Models & Color Image Processing

- Color fundamentals
- RGB and HSV color models
- Pseudo-color image processing
- Full-color image processing
- Color & Geometric transformations
- Digital image compositing
- Appendix:
    - The CMY and CMYK color models
    - Matte creation and manipulation in image compositing
    - Reading 1: *Compositing Digital Images*
    - Reading 2: *Alpha and the History of Digital Compositing*

Color Fundamentals

- Color is a powerful descriptor that often simplifies object identification and extraction from a scene.

  (Color is a visual feature which is immediately perceived when looking at an image.)

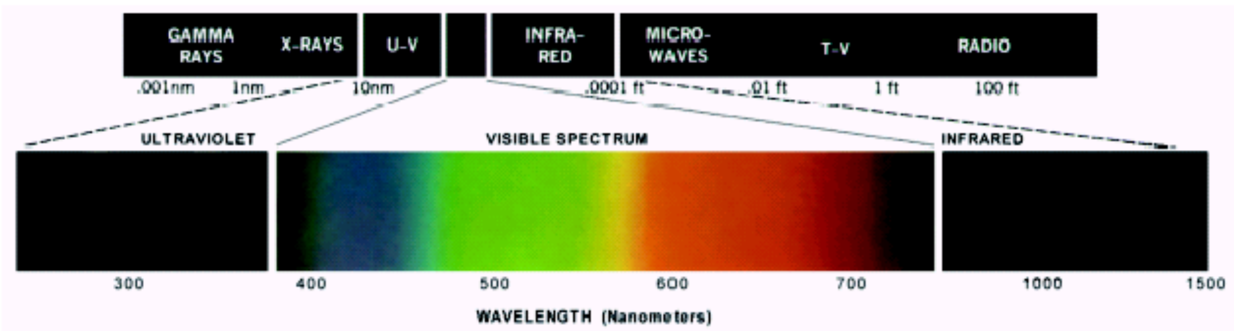
- Humans can discern thousands of color shades and intensities, compared to about only two dozen shades of gray.

Color Fundamentals

Light and the Electromagnetic (EM) Spectrum

- Light is a form of electromagnetic radiation
- The range of colors we perceive in visible light represents a very small portion of the electromagnetic spectrum.
- When view in full color, no color in the spectrum ends abruptly, but rather each color blends smoothly into the next.

Wavelengths comprising the visible range of the electromagnetic spectrum

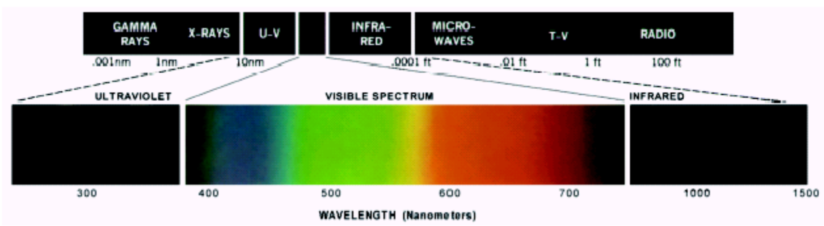


Color Fundamentals

- The *colors* that humans perceive in an object are determined by the nature of the light *reflected* from the object.

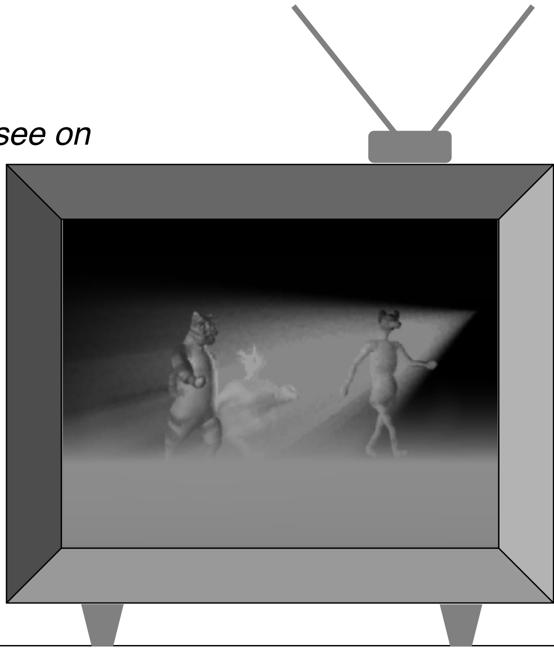
- A body that reflects light that is balanced in all visible wavelengths appears white to the observer. However, a body that favors reflectance in a limited range of the visible spectrum exhibits some shades of color.

- For example, red objects reflect light with wavelengths primarily in the 650 to 700 nm range while absorbing most of the energy at other wavelengths.



Color Fundamentals

- Light that is void of color is called *achromatic* or *monochromatic* light. The only attribute of such light is its *intensity*, or amount.
  - *Achromatic light is what viewers see on a black and white television set.*
- The term *gray level* refers to a scalar measure of intensity that ranges from black, to grays, and finally to white.



Color Fundamentals

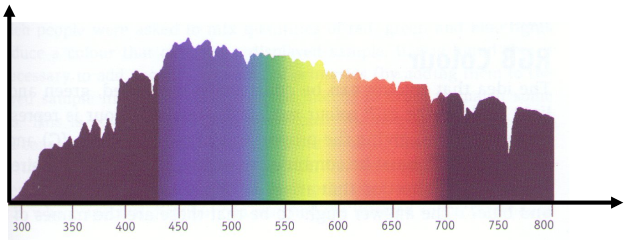
- Color is a subjective sensation produced in the brain.
- In order to reproduce color electronically, or manipulate it digitally, we need a model of color which relates that subjective sensation to measurable and reproducible physical phenomena.

Indoor scene: a house cat on a rug

Outdoor scene: naturally colored image of tiger in grass !

Color Fundamentals

- Since light is a form of electromagnetic radiation, we can measure its *wavelength* – the wavelength of visible light lies roughly between 400nm and 700nm – and its *intensity*.
- We can combine these measurements into a *spectral power distribution (SPD)*, a description of how the intensity of light from some particular source varies with wavelength.

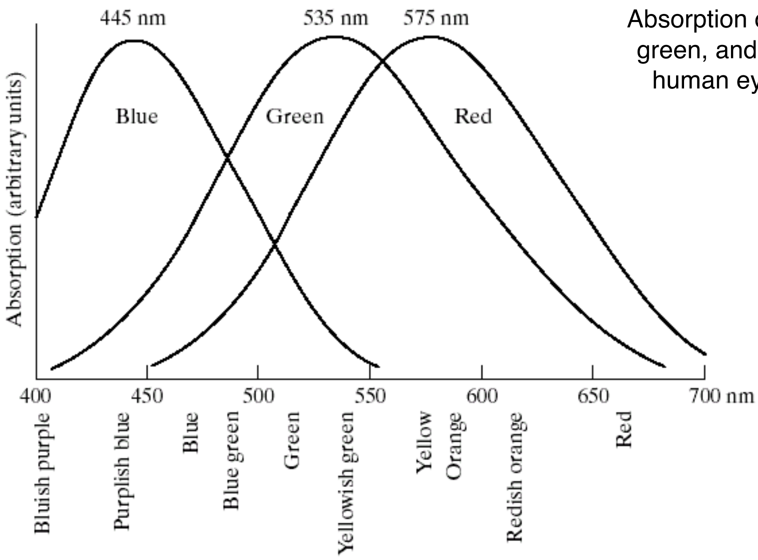


The SPD of daylight

- However, SPDs are too cumbersome to work with when we are specifying colors for use in computer graphics, so we need to adopt a different approach.

Color Fundamentals

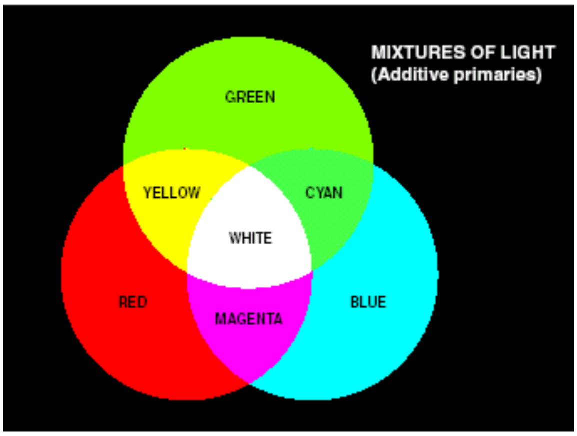
- The human eye contains two different sorts of receptor cells:
- *rods*, which provide night-vision and cannot distinguish color, and
- *cones*, which are highly sensitive to color and in turn come in three different sorts, which respond to different wavelengths of light.



Absorption of light by the red, green, and blue cones in the human eye as a function of wavelength

Color Fundamentals

- The fact that our perception of color derives from the eye's response to three different groups of wavelengths leads to the *tristimulus theory* – that any color can be specified by just three values, giving the weights of each of three components.
- We call **red**, **green** and **blue** the *additive primary colors*.



Color Fundamentals

- A color can be represented by three values. We can write this representation in the form (r,g,b), where r, g and b are the amounts of red, green and blue light making up the color. By "amount", we mean the proportion of pure (saturated) light of that primary.
- Examples:
  (100%, 0%, 0%) – pure saturated primary red;
  (50%, 0%, 0%) – a darker red;
  (100%, 50%, 100%) – mauve;
  (0%, 0%, 0%) – black;
  (100%, 100%, 100%) – white.

Color Models

- A *color model* (also called *color space* or *color system*) is a specification of a coordinate system and a subspace within that system where **each color is represented by a single point**.

- Most color models in use today are oriented either toward hardware (e.g. for color monitors and printers) or toward applications where color manipulation is a goal (e.g. in the creation of color graphics for animation).

Color Models

- The **RGB** (red, green, blue) Model
  - The RGB color model is the most important means of representing colors used in images for multimedia, because it corresponds to the way in which color is produced on computer color monitors, and it is also how color is detected by scanners.

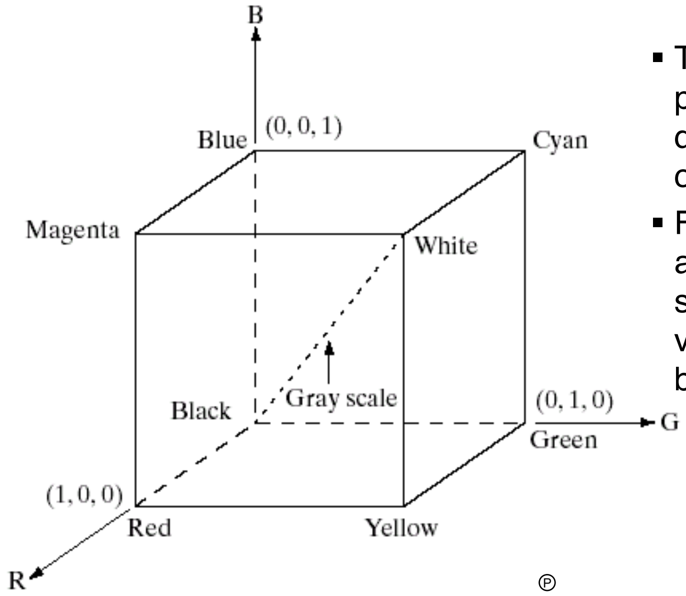
- The **CMYK*** (cyan, magenta, yellow, black) Model
  - Better for color printing

- The **HSV** (hue, saturation, value) (also called **HSB** or **HSI**)
  - It corresponds closely with the way humans describe and interpret color

* More details on the CMYK model can be found in this week's Appendix.

The RGB Color Model

- Each color appears in its primary spectral components of red, green, and blue.
- This model is based on a Cartesian Coordinate System.
- The color subspace of interest is the cube, in which RGB values are at three corners; cyan, magenta, and yellow are at three other corners; black is at the origin; and white is at the corner farthest from the origin.

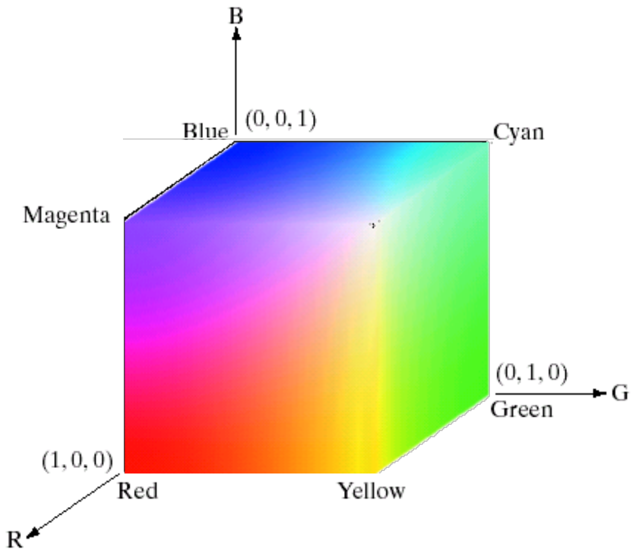


- The different colors in this model are points on or inside the cube, and are defined by vectors extending from the origin.
- For convenience, the assumption is that all color values have been normalized so that the cube is the unit cube, i.e., all values of R, G, and B are assumed to be in the range [0,1].

* In this model, the gray scale (points of equal RGB values) extends from black to white along the line joining these two points.

The RGB Color Model

- 256 is a very convenient number to use in a digital representation, since a single 8-bit byte can hold exactly that many different values, usually considered as numbers in the range 0 to 255. Thus, an RGB color can be represented in three bytes, or 24 bits.
- The number of bits used to hold a color value is often referred to as the *color depth*.
- The common color depths are sometimes distinguished by the terms *millions of colors* (24 bit), *thousands of colors* (16 bit) and *256 colors* (8 bit).



The RGB Color Model

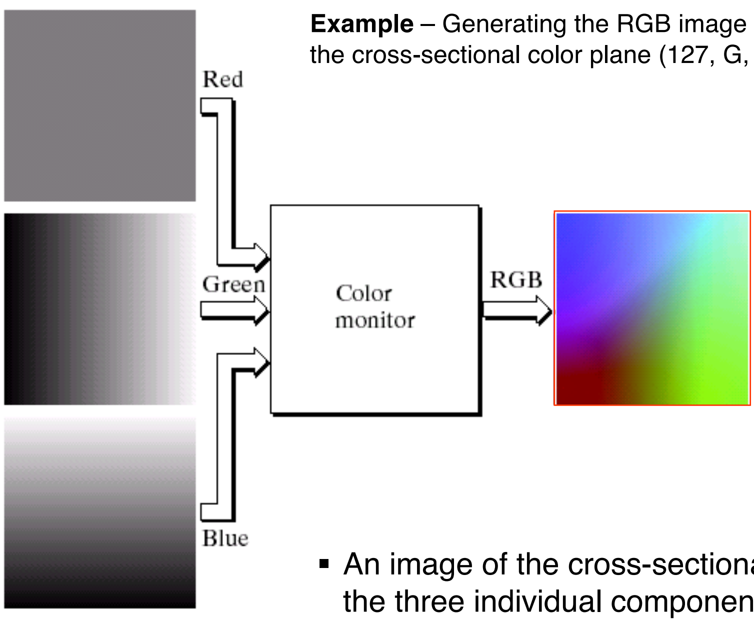
RGB 24-bit color cube

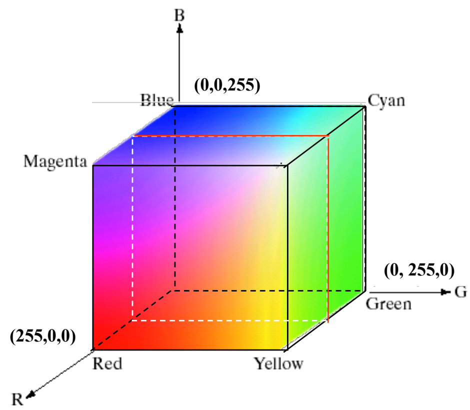
- The color cube is a solid, composed of the $(2^8)^3$ = 16,777,216 colors.

- A convenient way to view these colors is to generate **<u>color planes</u>** (faces or cross sections of the cube). This is accomplished simply by fixing one of the three colors and allowing the other two to vary.

The RGB Color Model

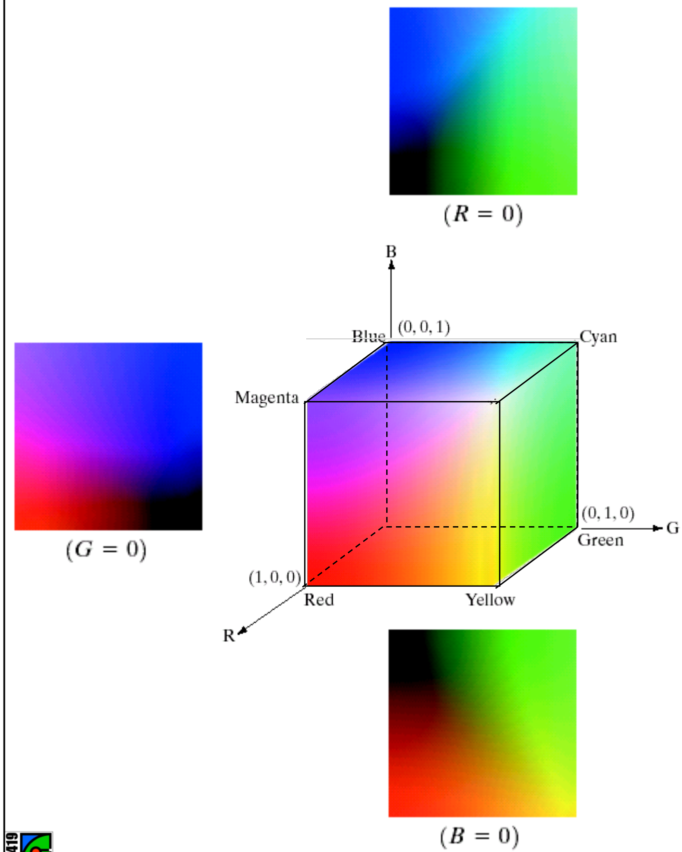
Example – Generating the RGB image of the cross-sectional color plane (127, G, B)

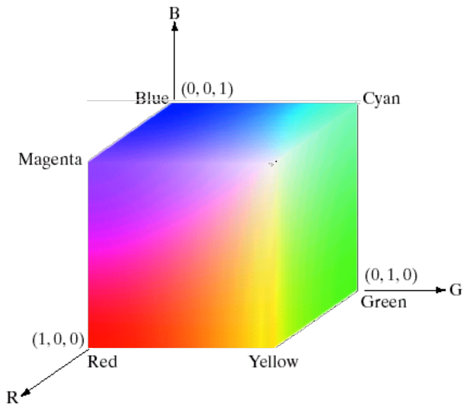




- An image of the cross-sectional plane is viewed simply by feeding the three individual component images into a color monitor.

- In the component images, 0 represents black and 255 represents white (note that these are gray-scale images).





Example – The three hidden surface planes of the color cube

The HSV Color Model

- The RGB and CMY color models are ideally suited for hardware implementations. In addition, the RGB system matches nicely with the fact that the human eye is strongly perceptive to red, green, and blue primaries.
- Unfortunately, the RGB, CMY and other similar color models are not well suited for **describing** colors in terms that are practical for human interpretation.
- When humans view a color object, we describe it by its **hue**, **saturation**, and **brightness** (value).

The HSV Color Model

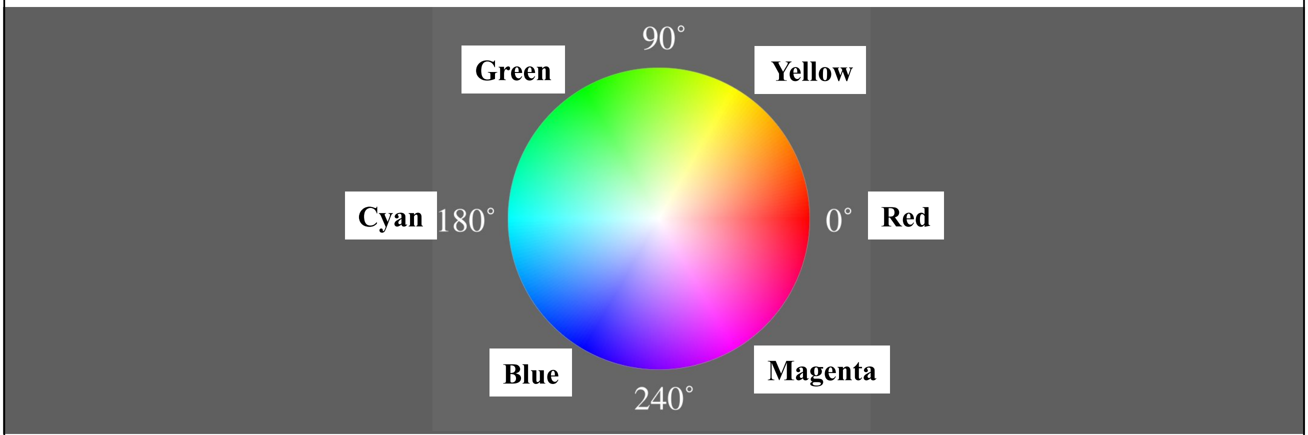
- A particularly useful alternate method for representing (and manipulating) the colors of an image is known as the **HSV color space**. "HSV" refers to the hue, saturation and value of a pixel*.
  - In many ways, HSV space is a much more intuitive method of dealing with color, since it uses terms that match more closely with the way a layperson talks about color.
  - When speaking of color conversationally, instead of characterizing a color as having 85% red, 0% green, and 90% blue, we would tend to say that the color is a "saturated magenta".
  - The HSV model follows this thinking process, while still giving the user precise definition and control.

----

* Variations on the HSV model include HSI and HSB, in which the third component is either Intensity or Brightness, respectively.

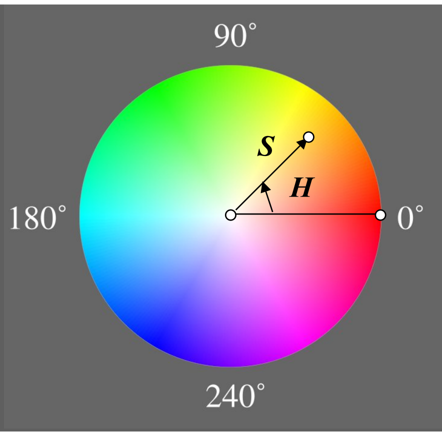
The HSV Color Model

The **hue** of a pixel refers to its basic color – such as red or yellow or violet or magenta. It is usually represented in the range of 0 to 360, referring to the color's location (in degree) around a circular color palette. For example, the color located at 90° corresponds to a yellow green.



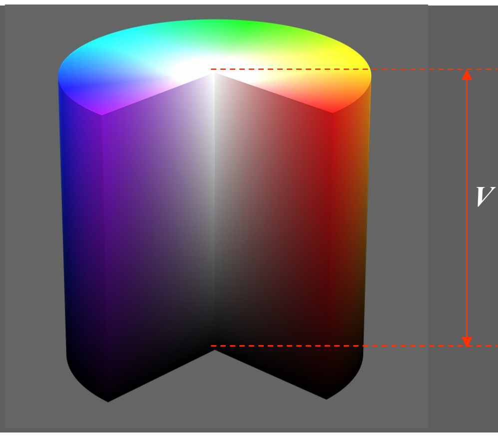
The HSV Color Model

<u>Saturation</u> is the brilliance or purity of the specific hue that is present in the pixel. If we look again at the HSV color wheel, colors on the perimeter are fully saturated, and the saturation decreases as you move to the center of the wheel.

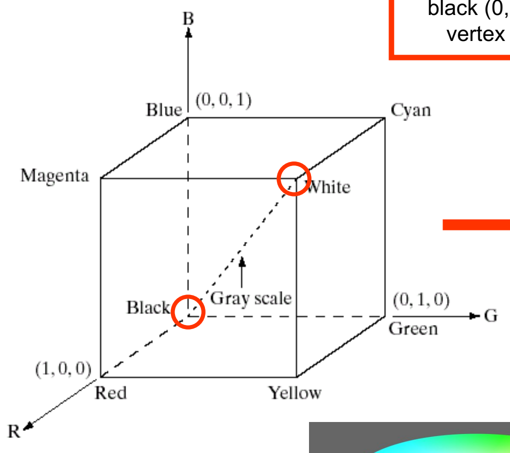


The HSV Color Model

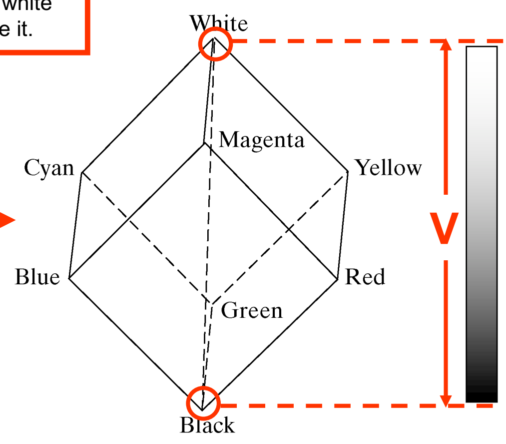
<u>Value</u> can just be thought of as the brightness of the color, although strictly speaking it is defined to be the maximum of red, green, or blue values. Trying to represent this third component means that we need to move beyond a 2D graph. The value is graphed along the third axis, with the lowest value, black, being located at the bottom of the cylinder. White, the highest brightness value, is consequently located at the opposite end.

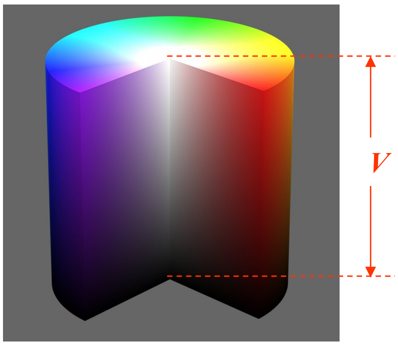


Conceptual relationships between the RGB and HSV color models



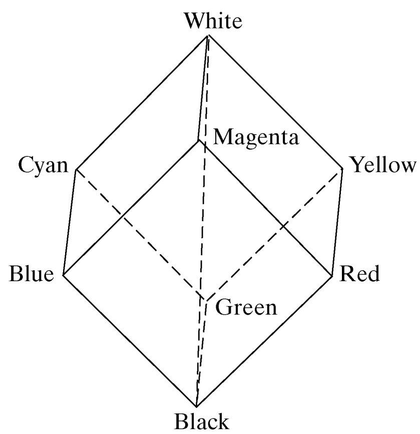
Take the color cube and stand it on the black (0,0,0) vertex, with the white vertex (1,1,1) directly above it.

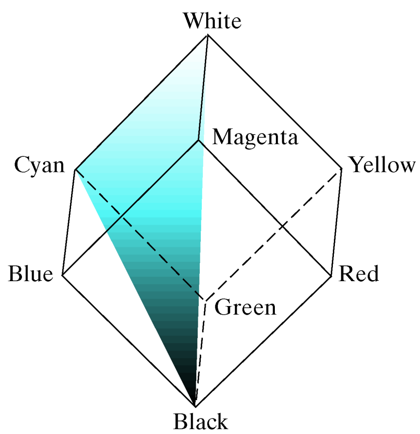




The intensity (gray scale) is along the vertical line joining these two vertices.

Conceptual relationships between the RGB and HSV color models

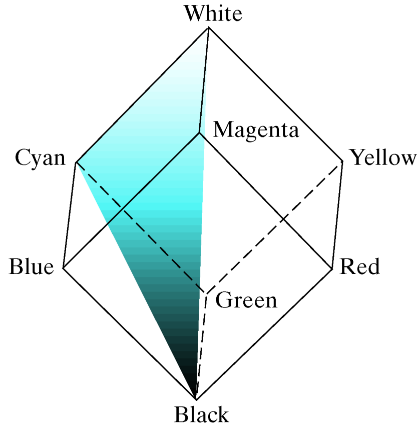




- An RGB color image can be viewed as three monochrome intensity images (representing red, green, and blue), so it should come as no surprise that we should be able to extract intensity from an RGB image. This becomes rather clear if we take the color cube and stand it on the black (0,0,0) vertex, with the white vertex (1,1,1) directly above it. Then the intensity (gray scale) is along the vertical line joining these two vertices. Thus, if we wanted to determine the intensity component of any color point, we would simply pass a plane perpendicular to the intensity axis and containing the color point. The intersection of the plane with the intensity axis would give us a point with intensity value in the range [0,1]. We also note with a little thought that the saturation (purity) of a color increases as a function of distance from the intensity axis. In fact, the saturation of points on the intensity axis is zero, as evidenced by the fact that all points along this axis are gray. (Con't)

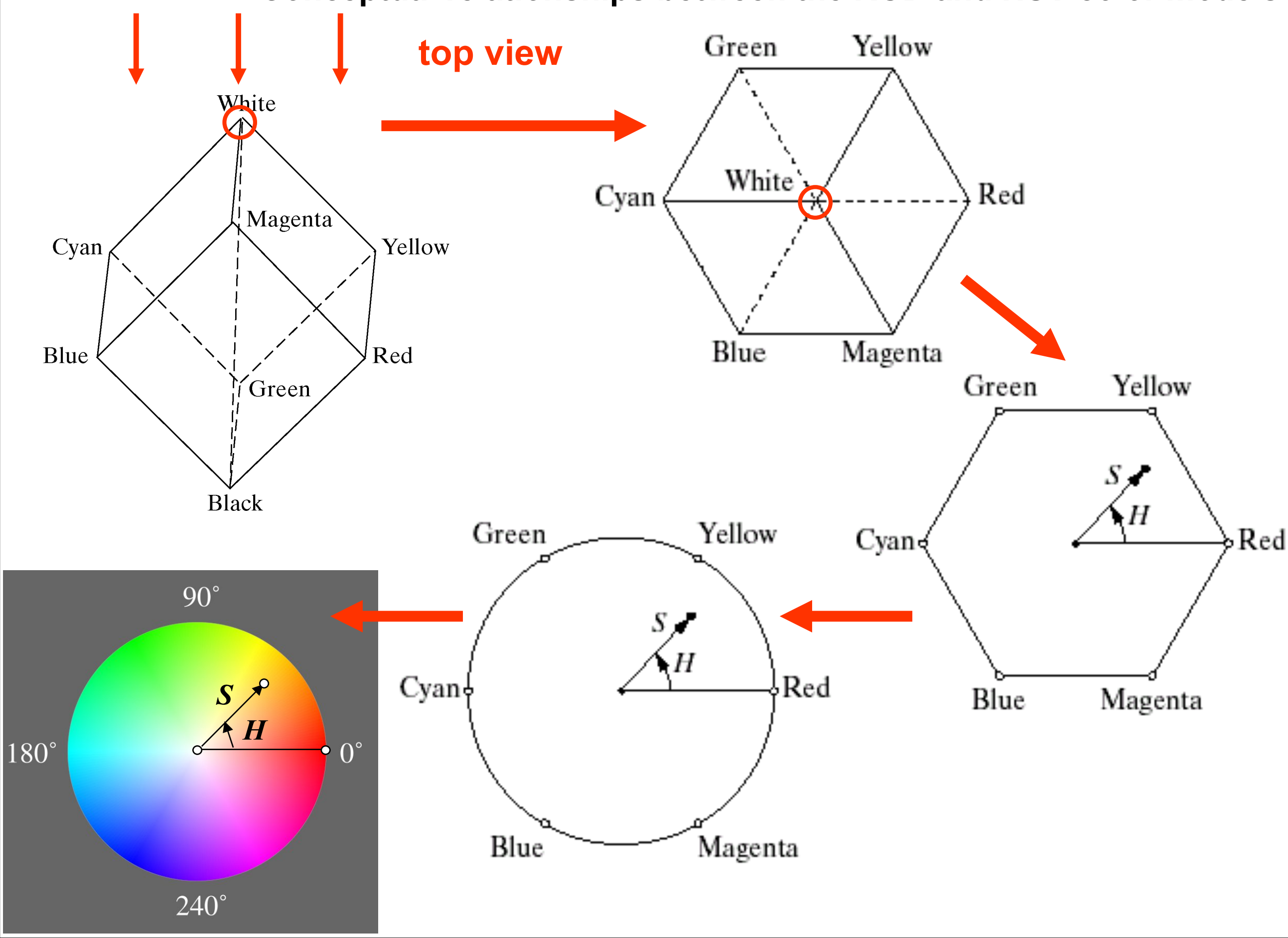
Conceptual relationships between the RGB and HSV color models

- In order to see how <u>hue</u> can be determined also from a given RGB point, please note the shaded plane defined by three points (black, white, and cyan). The fact that the black and white points are contained in the plane tells us that the intensity axis also is contained in the plane. Furthermore, we see that all points contained in the plane segment defined by the intensity axis and the boundaries of the cube have the same hue (cyan in this case). This is because the colors inside a color triangle are various combinations or mixtures of the three vertex colors. If two of those points are black and white and the third is a color point, all points on the triangle would have the same hue because the black and white components do not contribute to changes the hue (of course, the intensity and saturation of points in this triangle do change). By rotating the shaded plane about the vertical intensity axis, we would obtain different hues.

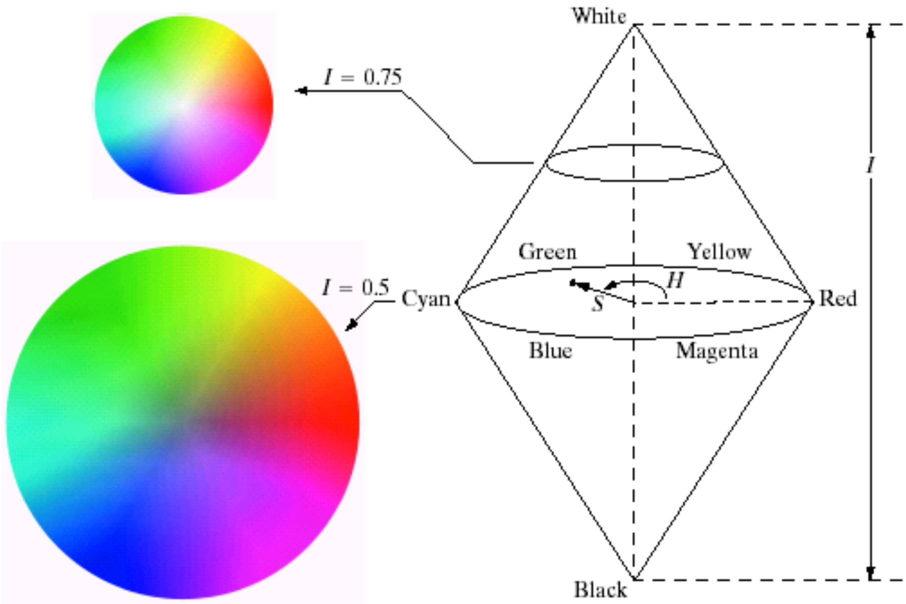


- From these concepts we arrive at the conclusion that the hue, saturation, and intensity values required to form the HSV space can be obtained from the RGB color cube. That is, we can convert any RGB point to a corresponding point in the HSV color model by working out the geometrical formulas.

Conceptual relationships between the RGB and HSV color models



The HSV Color Model based on Circular Color Planes



- The circles are perpendicular to the vertical intensity axis.

Color Manipulations – HSV manipulations

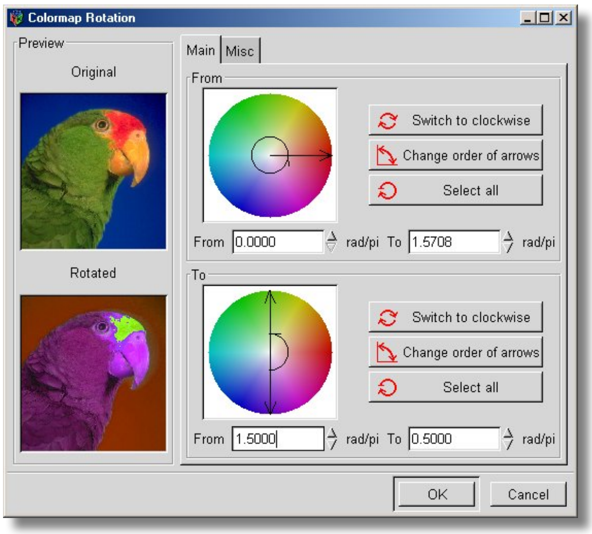
- Even though we have talked about the HSV color space as an alternate method of representing data, **it is generally not used to store images**. Rather, it is much more commonly employed as **a useful paradigm for manipulating colors**.
- For example, affecting the saturation of an image merely by adjusting RGB values can be rather cumbersome, but using the HSV model allows us to directly access the saturation component, making manipulation trivial.


The original image


50% saturation reduced image

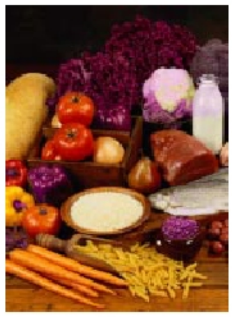
Color Manipulations – HSV manipulations

Because HSV represents hue in a circular fashion, we can now “rotate” all the colors in an image by a certain amount.



You can modify an image by rotating with 180° through the color spectrum. You will notice that every color in the image has been shifted to its complementary color, while still preserving the brightness and saturation relationships.

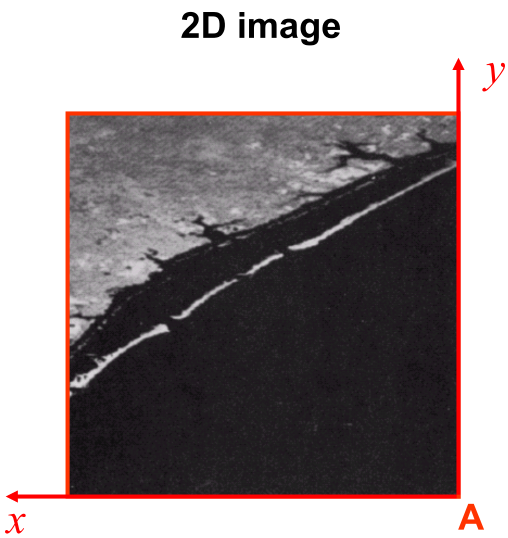
Original

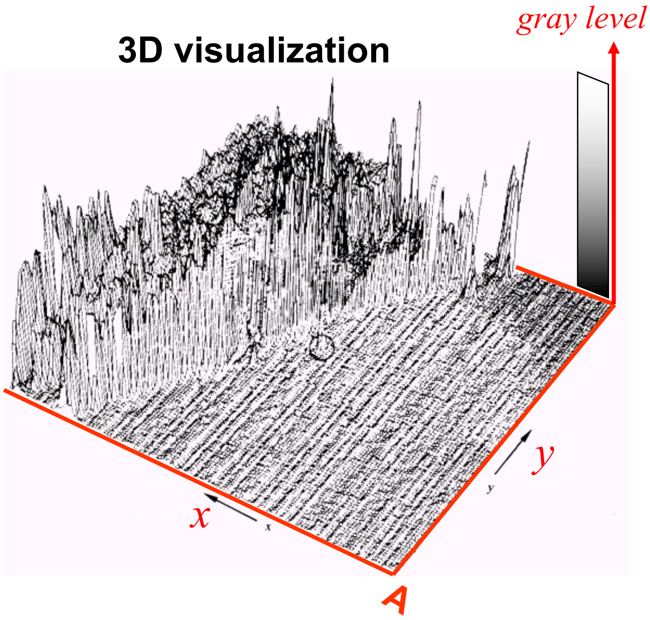


Rotated

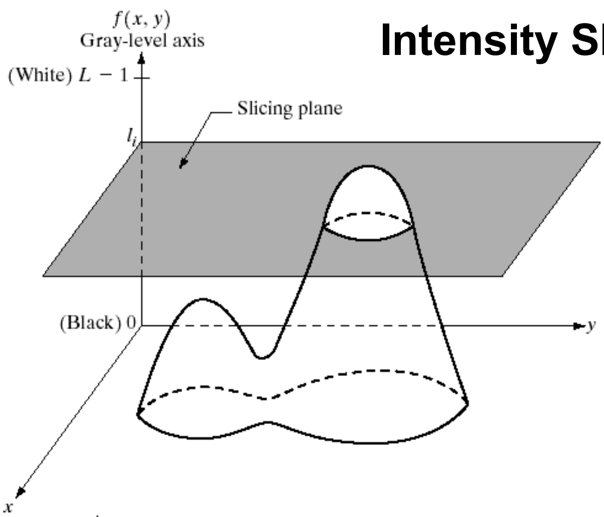
Pseudocolor Image Processing

- Pseudocolor (false color) image processing consists of **assigning colors to gray values** based on a specified criterion.
- The principal use of pseudocolor is for human visualization and interpretation of gray-scale events in an image or sequence of images.

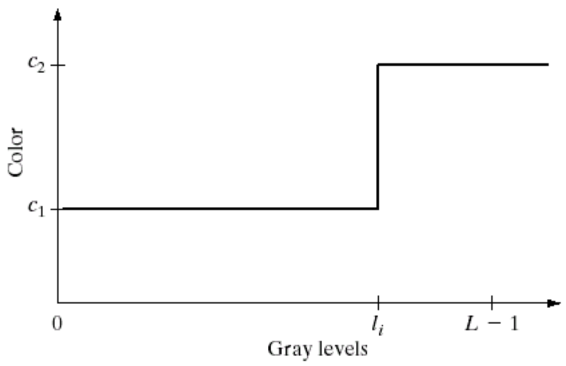




Intensity Slicing Coding

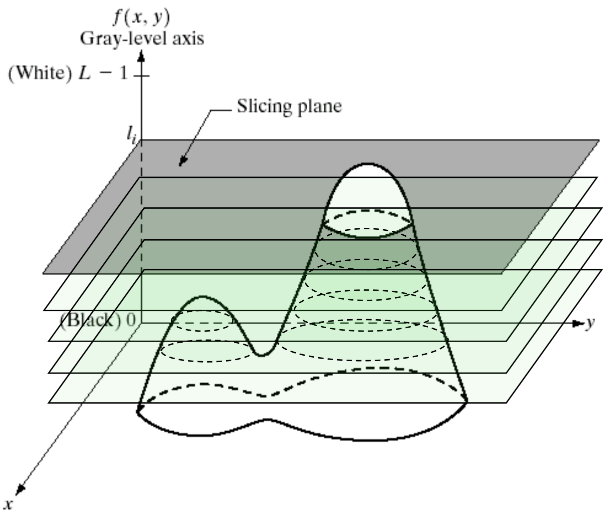


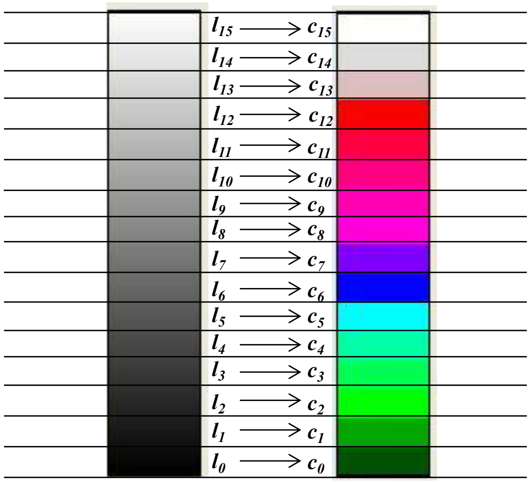
- If an image is interpreted as a 3-D function (intensity versus spatial coordinates), the method can be viewed as one of placing planes parallel to the coordinate plane of the image; each plane then "slices" the function in the area of intersection.
- If a different color is assigned to each side of the plane, any pixel whose gray level is above the plane will be coded with one color, and any pixel below the plane will be coded with the other.



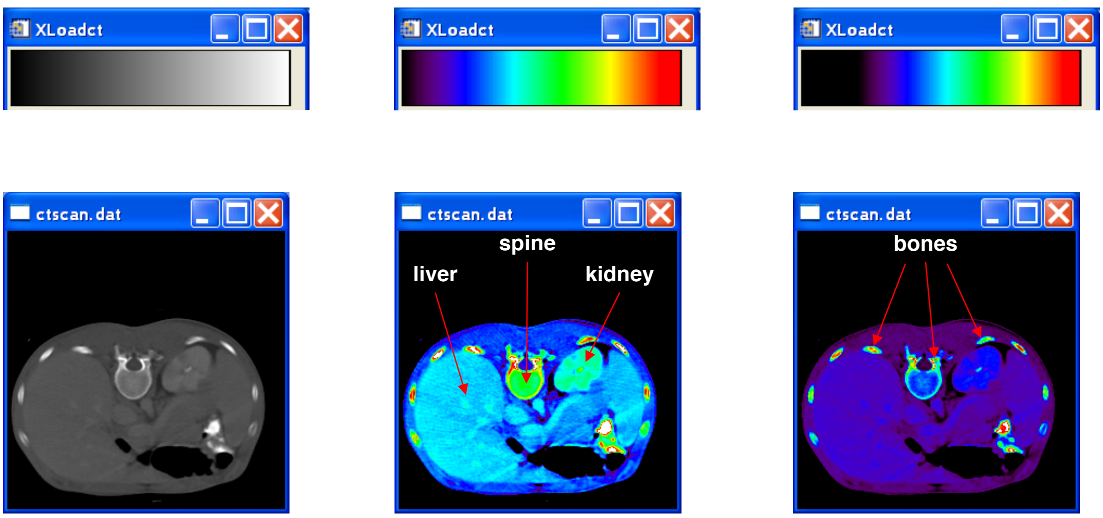
- An alternative representation of the intensity slicing technique

Intensity Slicing Coding



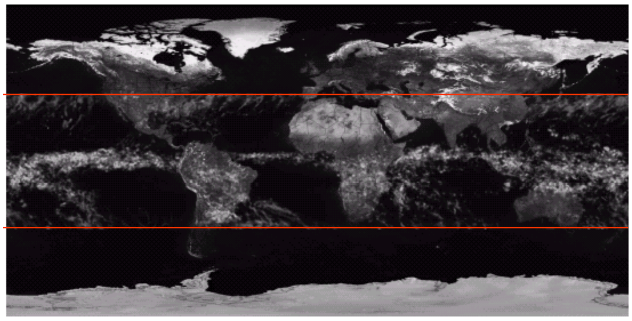


EXAMPLE_ Highlighting Region of interest (ROI) for CT images

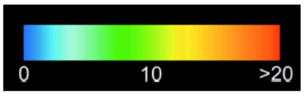


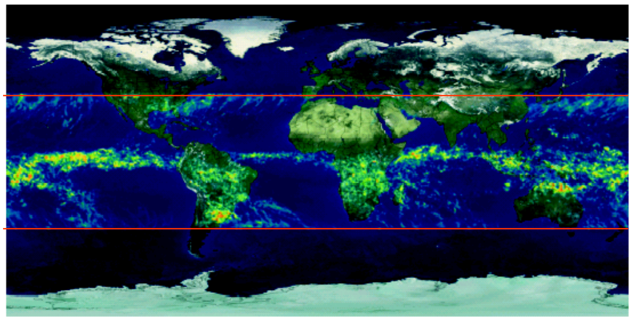
EXAMPLE_ The NASA TRMM (Tropical Rainfall Measuring Mission) satellite image

gray-scale image in which intensity corresponds to average monthly rainfall

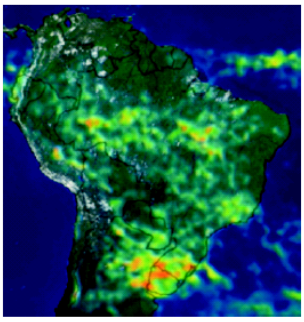


colors assigned to intensity values



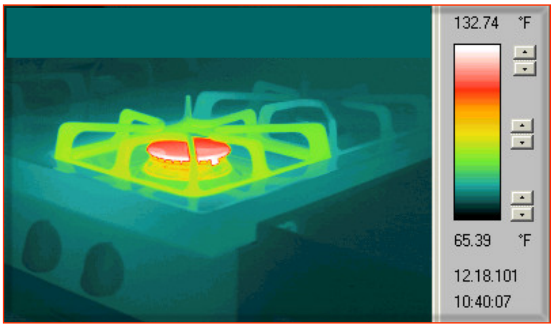


color-coded image

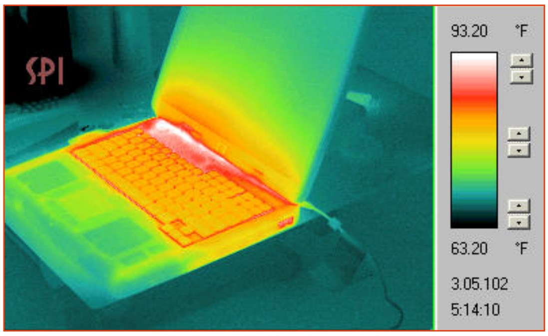


zoom of the South America region

EXAMPLE_ Imaging in the infra-red bands

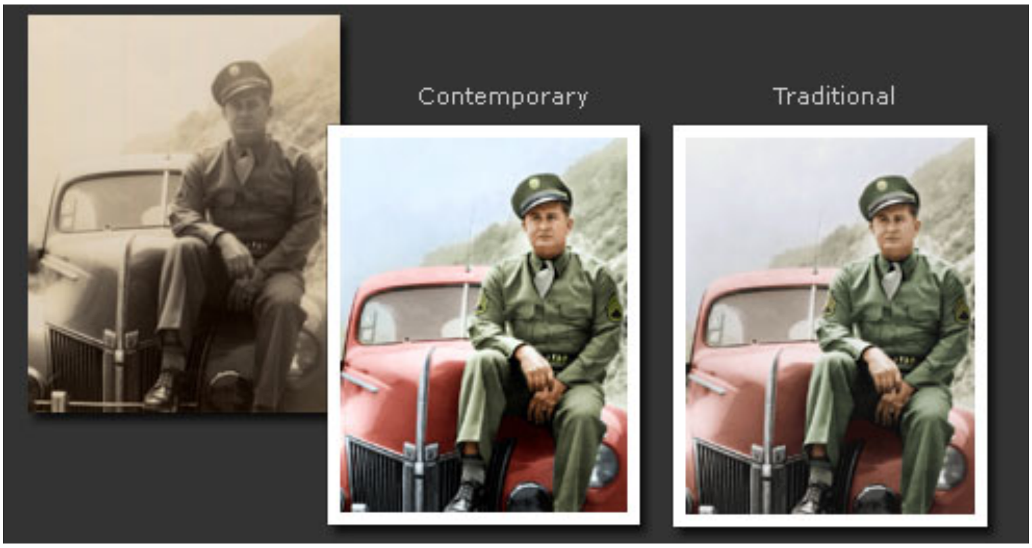


Stove top infra-red image

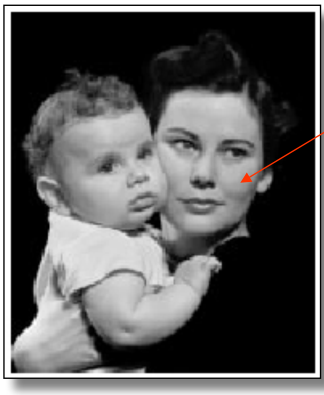


Laptop infra-red image

Pseudocolor Image Processing **Sample Colorize**

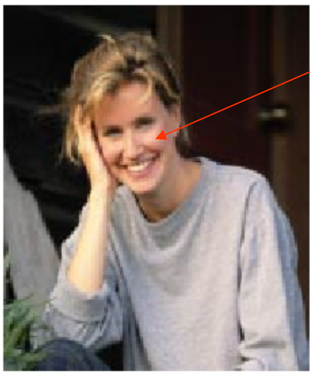


Pseudocolor Image Processing **Sample Colorize (from a color image)**



old image to colorize

select the part you want to colorize



source image

select the color source for skin tone

colorized image

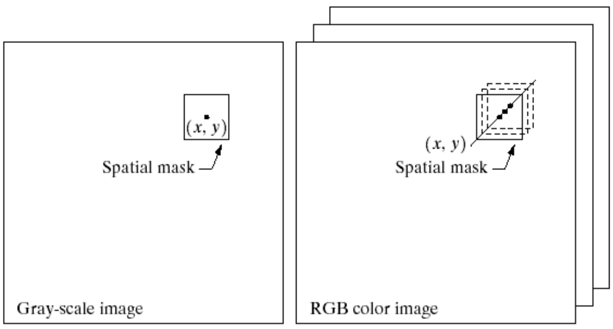
Pseudocolor Image Processing **Sample Colorize (from a gradient)**

old image to colorize

Full-Color Image Processing

Two major categories:

- Category 1: we process each component image individually and then form a composite processed color image from the individually processed components. (C1)
- Category 2: we work with color pixels directly. Because color images have at least three components, color pixels really are *vectors*. (C2)

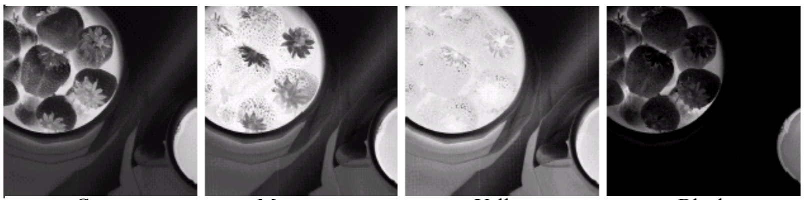


$$\mathbf{c}(x,y) = \begin{bmatrix} c_R(x,y) \\ c_G(x,y) \\ c_B(x,y) \end{bmatrix} = \begin{bmatrix} R(x,y) \\ G(x,y) \\ B(x,y) \end{bmatrix}$$

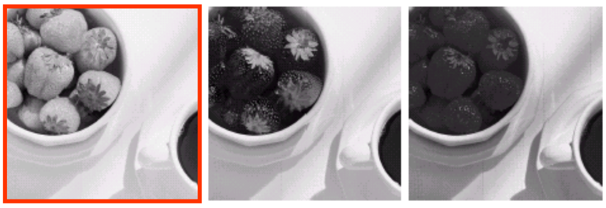
For an image of size M x N, there are MN such vectors, $\mathbf{c}(x,y)$, for $x = 0,1,2,\ldots,M-1$; $y = 0,1,2,\ldots,N-1$.

A full-color image and
its various color-space components

Full color



Cyan Magenta Yellow Black



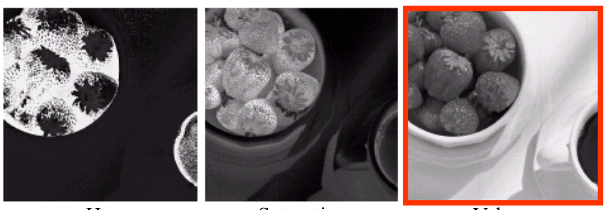
Red Green Blue

Hue Saturation Value

Example of C1 – HSV manipulation

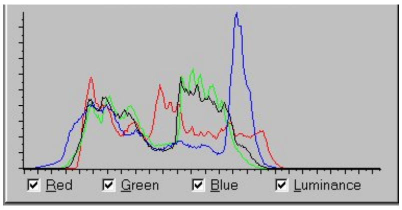
original image

Result of decreasing its intensity by 30%

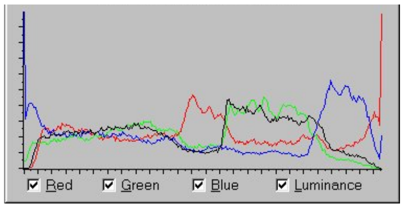


Hue Saturation Value

Example of C1 – Histogram equalization



Histogram
Equalized



Example of C2 – Spatial Filtering (for image sharpening)

Original image

Result of a typical
Laplacian filter mask

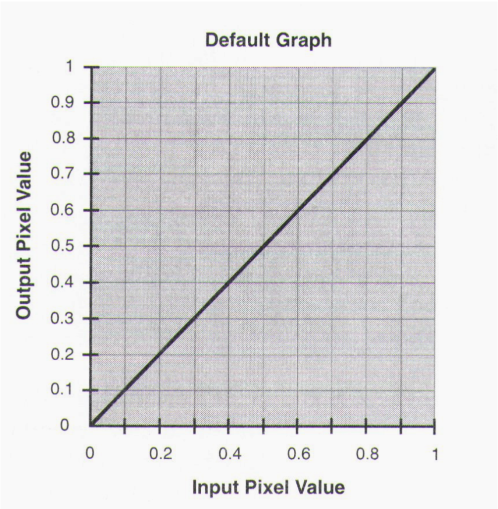
$$\begin{matrix} -1 & -1 & -1 \\ -1 & 9 & -1 \\ -1 & -1 & -1 \end{matrix}$$

A typical image
sharpening filter mask
used to implement the
linear *Laplacian* operator

Color Transformations

- Deal with processing the components of a color image within the context of a single color model.
- Color transformations can be performed on most desktop computers. In conjunction with digital cameras, flatbed scanners, and color printers, they turn a personal computer into a ***digital darkroom*** – allowing tonal adjustments and color corrections, etc.

Function / Parameter Curves



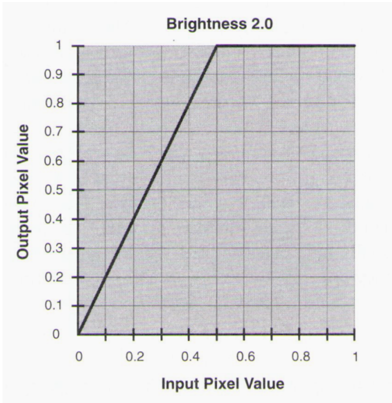
Graph of an unmodified image

The **function or parameter curves** are graphs that represent and control different attributes of an image, such as brightness or color. These attributes can be easily modified by manipulating the function curves without having to alter the image directly with a retouching tool. Making image manipulations that involve all of the image or large portions of it are best performed with function curves.

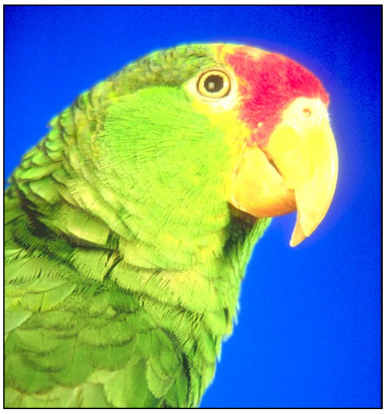
Function curves for image manipulation are usually represented by a line that starts at the lower left corner of a square and ends at the upper right corner. The straight diagonal line represents one or several untouched attributes of the original image. Any changes made to the line will result in changes to the image.

Function / Parameter Curves


The original image



Brightness 2.0 Graph

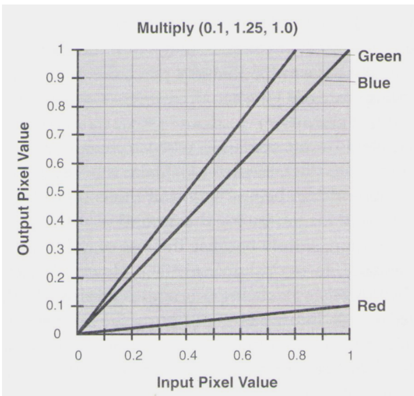


We can provide a brief equation to describe the operator. We usually treat the entire image as if it were a single variable. In these cases, we use the following conventions:

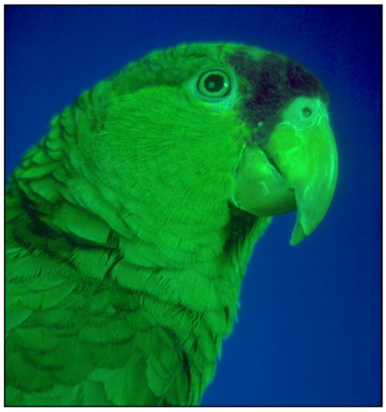
I = Input image; O = Output image. Thus, $O = I \times 2.0$ would refer to the above example, in which every pixel in the input image was multiplied by 2.0 to produce the output image.

Color Manipulations – RGB Multiply


The original image



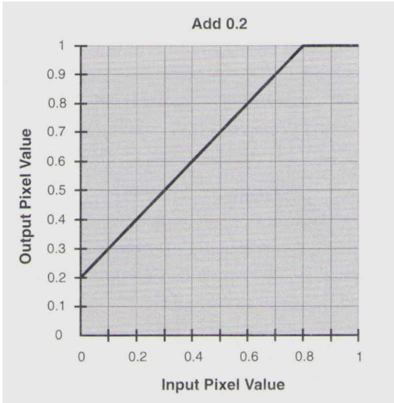
Graph of RGB multiplication



Most of the color-correction operators can be applied either to all the channels of an image equally or to individual channels in varying amounts. When applied equally, the result will tend to be an overall brightness or contrast modification. When different amounts are applied to different channels, a visual color shift will usually take place as well.

Color Manipulations – Add

The original image

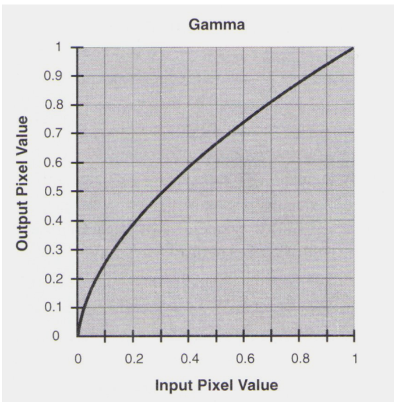


Graph of add 0.2 operation

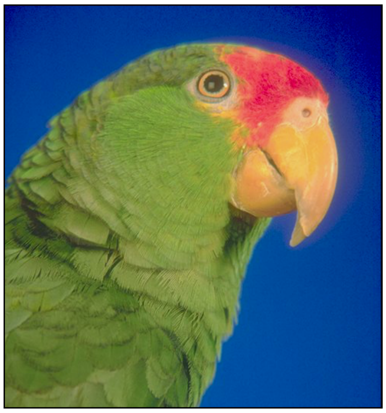
Instead of affecting the apparent brightness of an image by multiplying, we can add (or subtract) a constant value from each pixel. Notice that, unlike the multiplication operation (which keeps the deepest blacks at the same level), the blacks in this example have gone to gray. ($O = I + 0.2$)

Color Manipulations – Gamma Correction

The original image



Graph of gamma correction of 1.7



It uses an exponential function: $O = I^{1/Gamma}$. In other words, we raise the value of each pixel to power of 1 divided by the gamma value supplied.

Color Manipulations – Gamma Correction

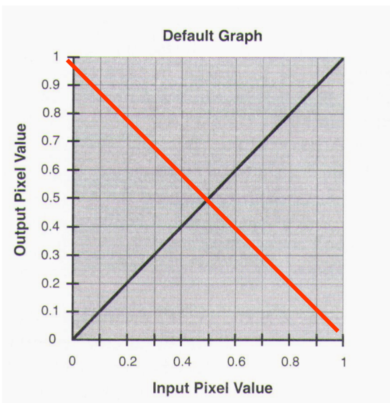
The reason why the Gamma operator is so popular becomes apparent when you examine what happens when you raise 0 to any power – it stays at 0*. A similar thing happens when 1.0 is raised to any power – it stays at 1.0. In other words, no matter what gamma correction you apply to an image, pixels with a value of 0 or 1.0 will remain unchanged. The only effect that the gamma operator has will be on non-black and non-white pixels. It makes the image tend to look more natural.

----

* The exception occurs when you raise anything to the zero-th power, which is defined as producing a result of 1. However, since applying a gamma value of 0 to $O = I^{1/Gamma}$ would produce a divide-by-zero problem, the generally accepted convention is that applying a gamma of 0 to an image will produce a black frame.

Color Manipulations – Invert


The original image

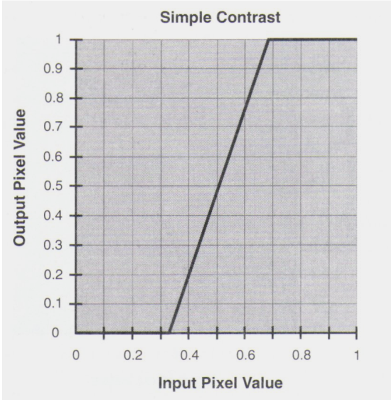


Graph of invert operation (red line)

An extremely simple operator: $O = (1 - I)$. Every pixel is replaced by the value of that pixel subtracted from 1.0. The result is an image that appears to be similar to the photographic negative of the original.

Color Manipulations – <u>Contrast</u>

The original image

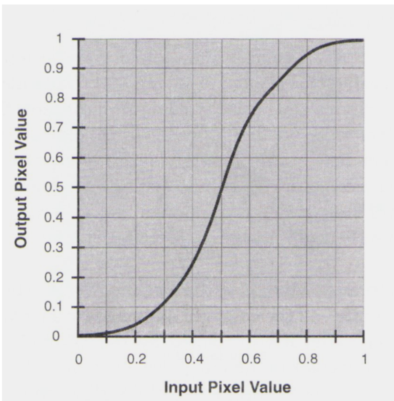


<u>Simple contrast operation</u>

A simple contrast could be implemented using a combination of the “Multiply” and “Subtract”. If we want to increase the contrast of an image, we could first subtract a constant value and then multiply the result by another constant, such as: $O = (I – 0.33) \times 3$).
However, contrast as applied in this manner is a less than ideal operator, since both the low end and the high end treat the image’s data rather harshly.

Color Manipulations – <u>Contrast</u>

The original image

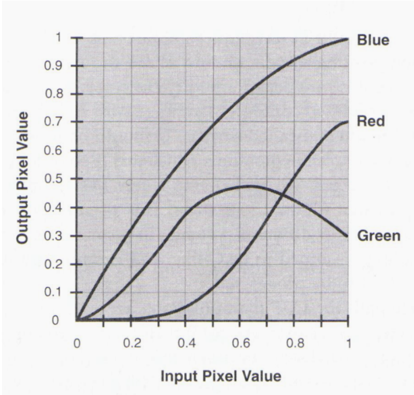


<u>Smoother contrast graph</u>

A better system is to apply gamma-like curves to the upper and lower ranges. This method tends to give a much cleaner image, particularly at the low and high ends of the brightness spectrum.

Color Manipulations – LUT Manipulations

The original image



Graph of arbitrary LUT curves

An input-to-output mapping is known as a **look-up table** (**LUT**). Such LUT-manipulation can give an extremely fine amount of control, even allowing specific color corrections across a narrow range of values. Here the output image has had all three channels modified by the set of three user-defined curves.

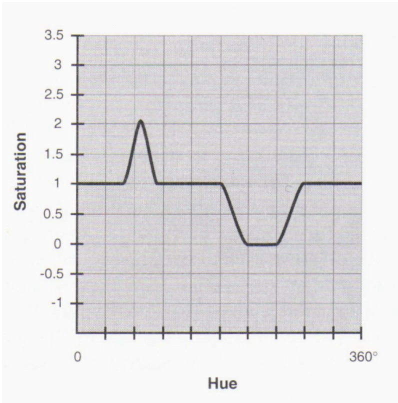
Color Manipulations – LUT Manipulations

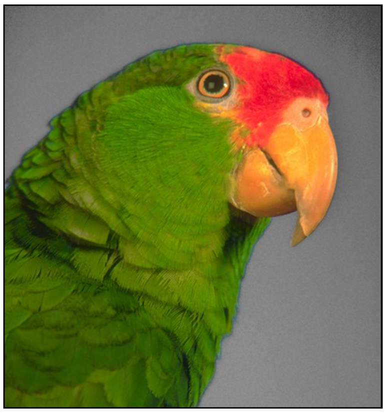
The LUT method of using user-defined curves to manipulate the colors of an image is not limited to merely modifying the brightness of certain regions in a given channel.

Given a flexible enough software package, one can choose to arbitrarily modify one channel based on another channel. This operation can be particularly useful if you are able to include HSV color space in the mix.

Color Manipulations – LUT Manipulations

The original image





Graph of hue versus saturation operation

We may wish to modify saturation based on a particular hue. Here the X-axis maps out the full range of hues available in an image, while the Y-axis controls the relative saturation. The center of the Y-axis is defined to be the point where saturation is unchanged. Pulling the curve upward will increase saturation, while moving it downward will decrease it. We have therefore chosen to de-saturate anything that is blue, while we supersaturate a narrow range of green values.

Image Restoration and Retouching

- As in image enhancement, the ultimate goal of *image restoration and retouching* techniques is to improve an image in some predefined sense.
- Image enhancement is largely a subjective process, while image restoration and retouching are for the most part an objective process, such as *geometric transformation*.

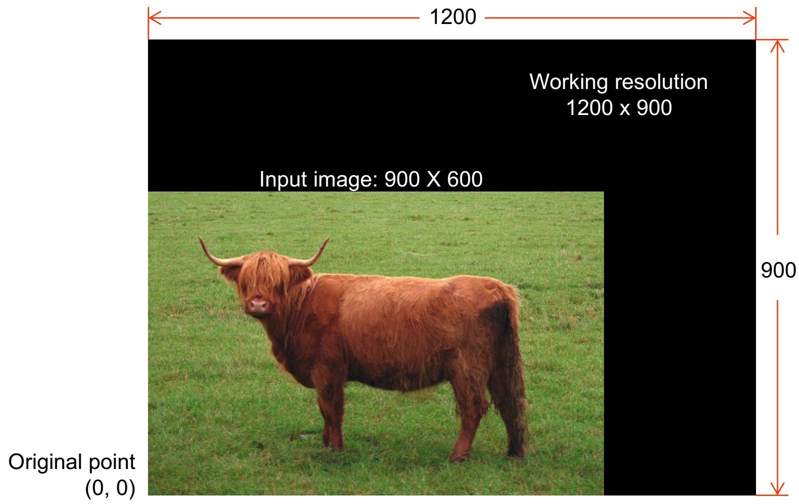
- Geometric transformations modify the spatial relationships between pixels in an image. Such transform operation causes some or all of the pixels in a given image to change their existing location.
  - Geometric transformations often are called *rubber-sheet transformations*, because they may be viewed as the process of “printing” an image on a sheet of rubber and then stretching this sheet according to some predefined set of rules.

Geometric Transformations

- Geometric transformations can be used for retouching images to obtain effects that would otherwise be impossible.
- Such effects include panning, rotating, scaling, warping, and various specialized distortion effects.

Special effect – “The Boils” in *MATRIX*

Geometric Transformations



A **working resolution** is typically the resolution of the image that will be produced once we are finished with our compositing operations.

Geometric Transformations – Panning

If you wish to apply a simple translation to the image, by offsetting it in both *X* and *Y*. Such a translation is usually referred to as a **pan**.

The image panned by 150 pixels along both axes.

Geometric Transformations – Panning

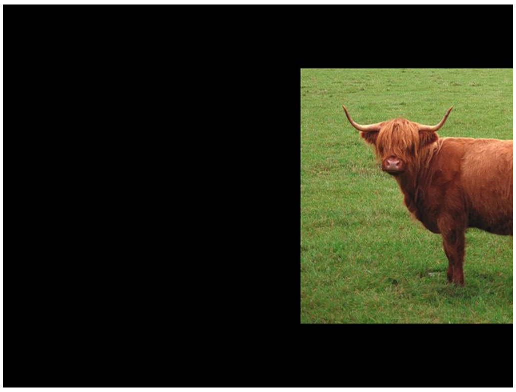


Image panned off-screen

Image panned with wrapping

If we move the input image 700 pixels along x-axis, it will cause part of the image to be moved beyond the borders of our working resolution. On most systems, the rest of the image will be cropped, or discarded. Some systems can allow the off-screen information to be preserved so that it can later be brought back into frame if needed.

Geometric Transformations – Rotation

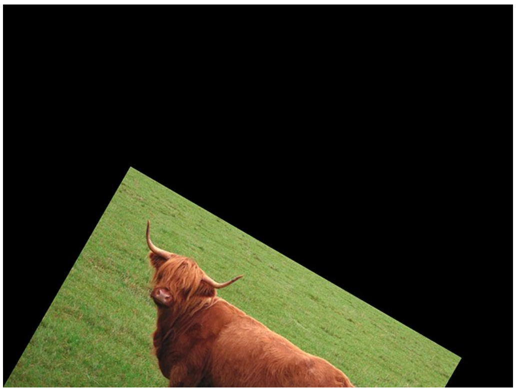


Image rotated 30˚ clockwise about the origin

Image rotated 30˚ clockwise about its approximate center

Geometric Transformations – Scale

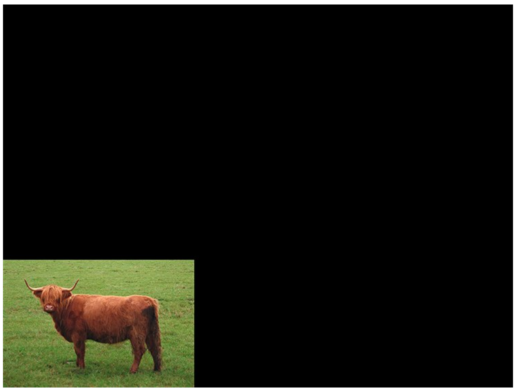


Image scaled by 50% around the origin

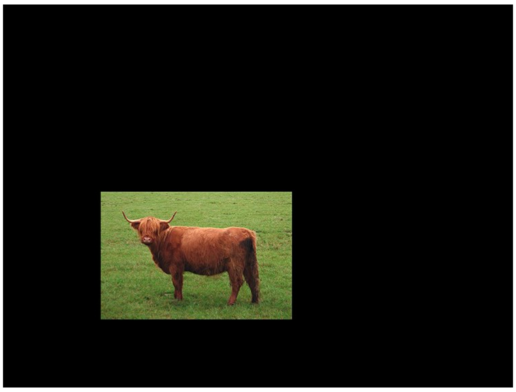
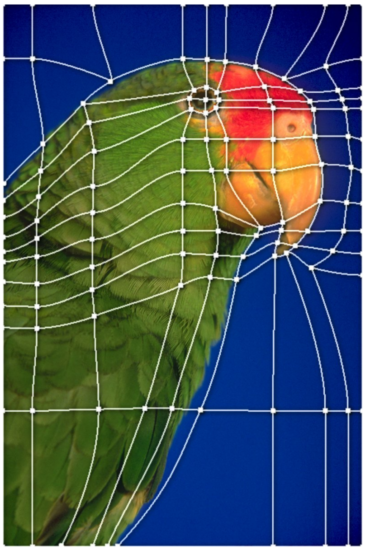


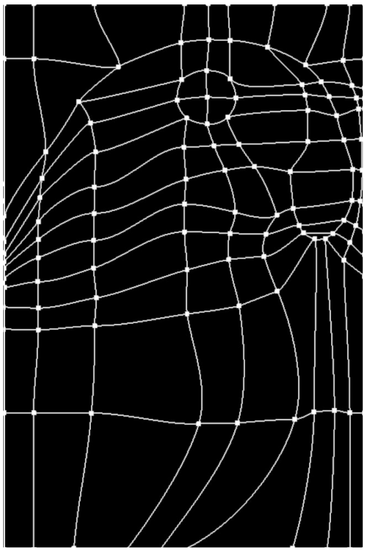
Image scaled by 50% around its approximate center

Geometric Transformations – Warping

An even more sophisticated method of distorting an image is known as **warping**. Conceptually, it is easiest to think of warping as if your image were printed on a thin sheet of flexible rubber. This rubber sheet can be pushed and pulled by various amounts in various areas until the desired result is obtained. Image warping is usually controlled by either a grid mesh or a series of splines.

Geometric Transformations – Warping


Original image with grid laid over it


Warping - Distorted grid


Resulting distorted image

Geometric Transformations – Warping

Warping operation performed using a sine function
$Y = Y + 50 \times \sin(X \times 0.02)$

Image Compositing

- **Image compositing** consists of combining two or more different images into one in such a way that an illusion of time and space is created. It seems that all the images happened at the same time and place and were recorded together.
  - When created with traditional tools – such as scissors, glue, and paper – image compositing results in a collage, which is an assembly or composition of image fragments or materials from different sources. In the film industry, image compositing is also known as matting because of the masks, or mattes, used in the compositing process.
- One of the main purposes of image compositing is usually to save expensive production costs or to simulate something that is physically impossible to create in our reality.

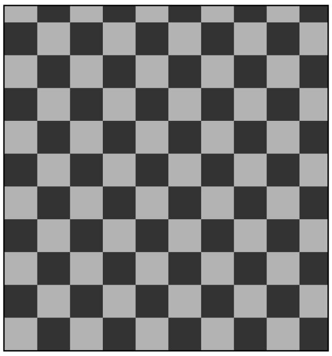
Image Compositing

- The process of compositing images from different sources into a single visually coherent image can be performed on both still and moving images. Still composites are often called collages, while moving composites result in dynamic composites or transition effects.
- Combining several shots from different sources into a single still or sequence is at the heart of all special visual effects, and also of many avant-garde artistic movements – such as Surrealism – that seek to subvert our notions of reality.

Basic Image Compositing


(a) Foreground image

+


(b) Background image

=

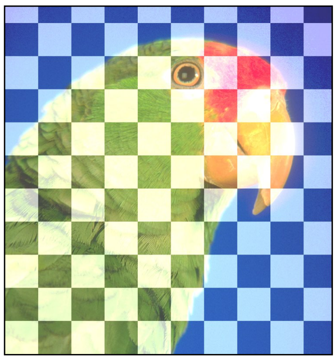


Image resulting from adding (a) and (b)

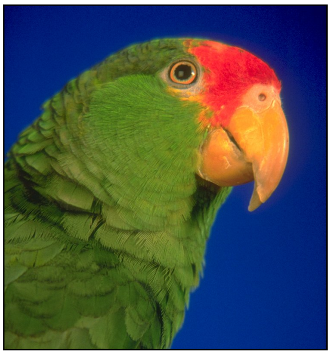
Example – **multisource operator ("ADD").** While this effect is certainly useful in a variety of situations, it does not give us the impression that any sort of layering has occurred. There is no sense that certain portions of one image are actually occluding the second image. To accomplish this, we need to introduce the concept of a matte.

The Matte Image

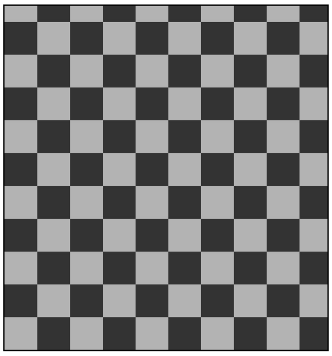
- **<u>Mattes</u>** are used during compositing when we only wish a portion of a certain image to be included in the result.


- Mattes are generally considered to be <u>single-channel, grayscale images</u>. There is no need for three separate channels, as there is when specifying color, since the transparency for any given pixel can be described by a single numerical value <u>in the range of 0 to 1</u>.
- A matte can also be bundled along with a three-channel color image as a discrete fourth channel. When the matte image is part of a four-channel image, it is known as the ***<u>matte channel</u>*** or the ***<u>alpha channel</u>***.

The Matte Image


(a) Foreground image


(c) Matte image


(b) Background image

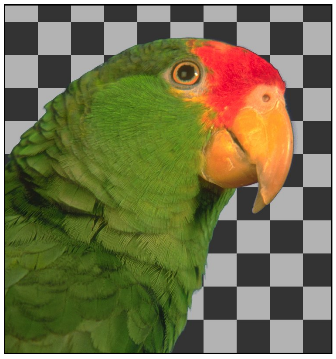
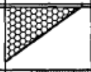
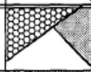
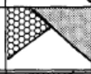
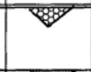
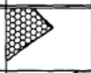
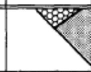
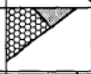
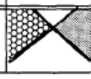


Image resulting from (a), (b) and (c)

The areas that are <u>white</u> in the matte channel are used to specify that the corresponding area of the foreground image is kept at full opacity. Conversely, the <u>black</u> areas of the matte are used to specify that the corresponding pixels in the foreground image will be transparent, or effectively removed, when it is placed over the background.

12 Compositing Operations*

operation	quadruple	diagram	F_A	F_B
clear	(0,0,0,0)		0	0
A	(0,A,0,A)		1	0
B	(0,0,B,B)		0	1
A **over** B	(0,A,B,A)		1	$1-\alpha_A$
B **over** A	(0,A,B,B)		$1-\alpha_B$	1
A **in** B	(0,0,0,A)		α_B	0
B **in** A	(0,0,0,B)		0	α_A
A **out** B	(0,A,0,0)		$1-\alpha_B$	0
B **out** A	(0,0,B,0)		0	$1-\alpha_A$
A **atop** B	(0,0,B,A)		α_B	$1-\alpha_A$
B **atop** A	(0,A,0,B)		$1-\alpha_B$	α_A
A **xor** B	(0,A,B,0)		$1-\alpha_B$	$1-\alpha_A$

*See this week's Appendix Reading 1

Multisource Operator - OVER

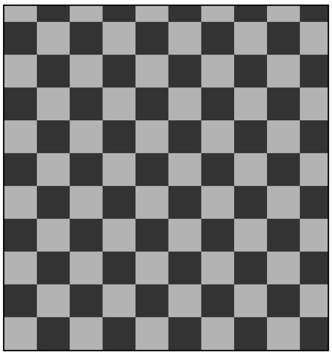
- The **Over** operator takes two images and, using a third image as a controlling matte, lays a portion of the first image on top of the second.

- Mathematically, when we place image **A** (the foreground) over image **B** (the background), using image **M** as the matte for image **A**, our output image **O** is as follows:

$$\boxed{\mathbf{O = (A \times M) + [(1 - M) \times B]}}$$

Multisource Operator – OVER

× (a) Foreground image **A**



× (b) Background image **B**

+

(c) Matte image **M**

(d) Inverted matte image **(1 – M)**

=

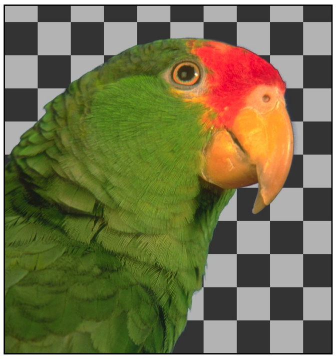
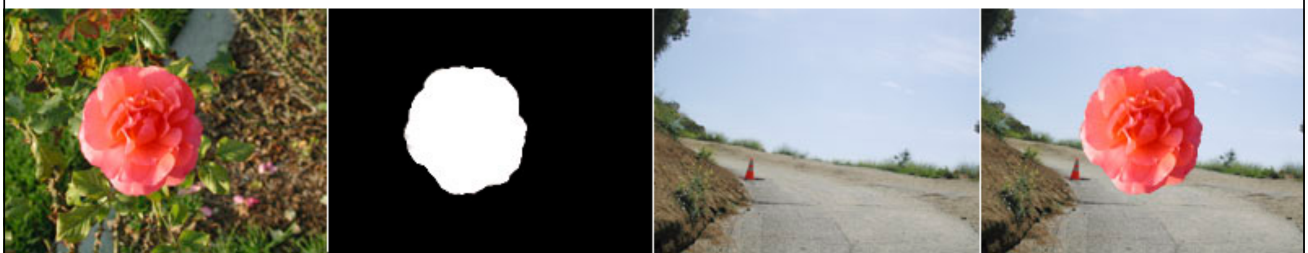


Image **O** resulting from placing (a) over (b) using the matte image in (c).

$$O = (A \times M) + [(1 - M) \times B]$$

hold-out matte or hold-back matte

Example



Multisource Operator - MIX

A **mix** is the weighted, normalized addition of two images. In other words, the two images are averaged together, often with one of the images contributing a larger percentage to the output.

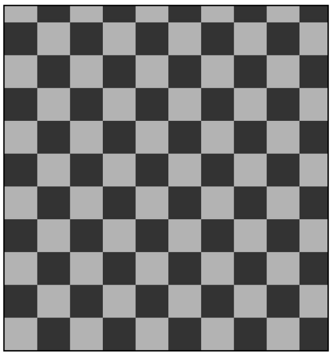
The equation for such a mix, where “MV” refers to the mix value (the percentage of the first image that we will be using), is as follows:

$$\boxed{O = (MV \times A) + [(1 - MV) \times B]}$$

Multisource Operators - MIX

(a) Foreground image **A**
× 75%

+



(b) Background image **B**
× 25%

=

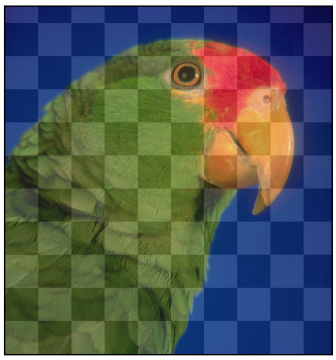


Image **O** resulting from mixing 75% of (a) with 25% of (b).

$$O = (MV \times A) + [(1 - MV) \times B]$$

*** MV = 75%**

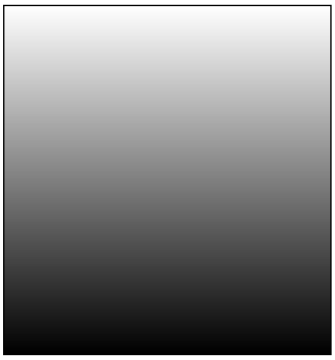
Masks

There are times when we wish to limit the extent of a certain operator's effect. A particularly useful method for doing this involves the use of a separate matte as a control image. Whenever a matte is used in this fashion, it is usually referred to as a **mask**.

Masks

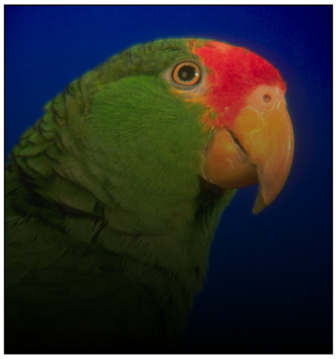
(a) Test image

×



(b) Masking image

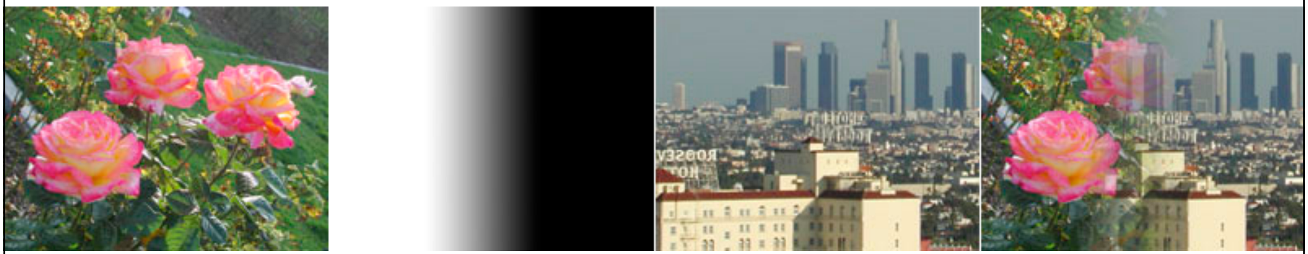
=



Test image (a) after being color corrected using (b) as the mask

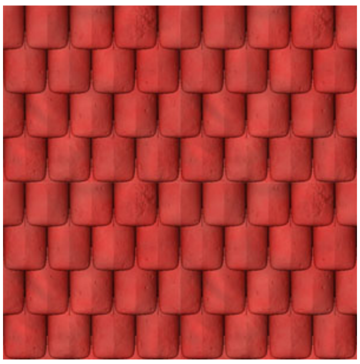
In the areas where the mask was completely solid, the resulting pixels remained unchanged. Where the mask was transparent, or black, the full attenuation could be applied. In any areas where the mask had intermediate gray values, the brightness was applied by a proportionally smaller amount.

Example

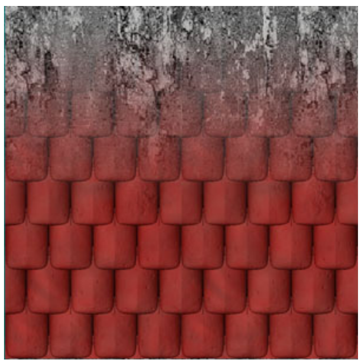


Example

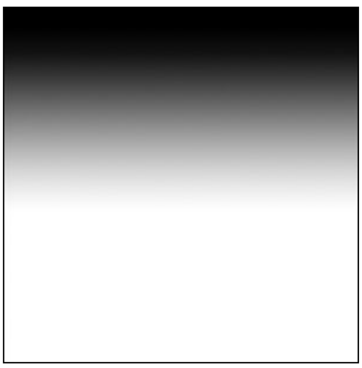
(b) A grunge map



(a) A brand-new roof image generated by CG



(d) Final layout



(c) A gradient mask

Week03 • Semester 2 • 2018

Color Models & Color Image Processing

Appendix

- ❑ The CMY and CMYK color models
- ❑ Matte creation and manipulation in image compositing
- ❑ Reading 1: *Compositing Digital Images*
- ❑ Reading 2: *Alpha and the History of Digital Compositing*