Computer Vision

Lecture 2: From human to the computer vision

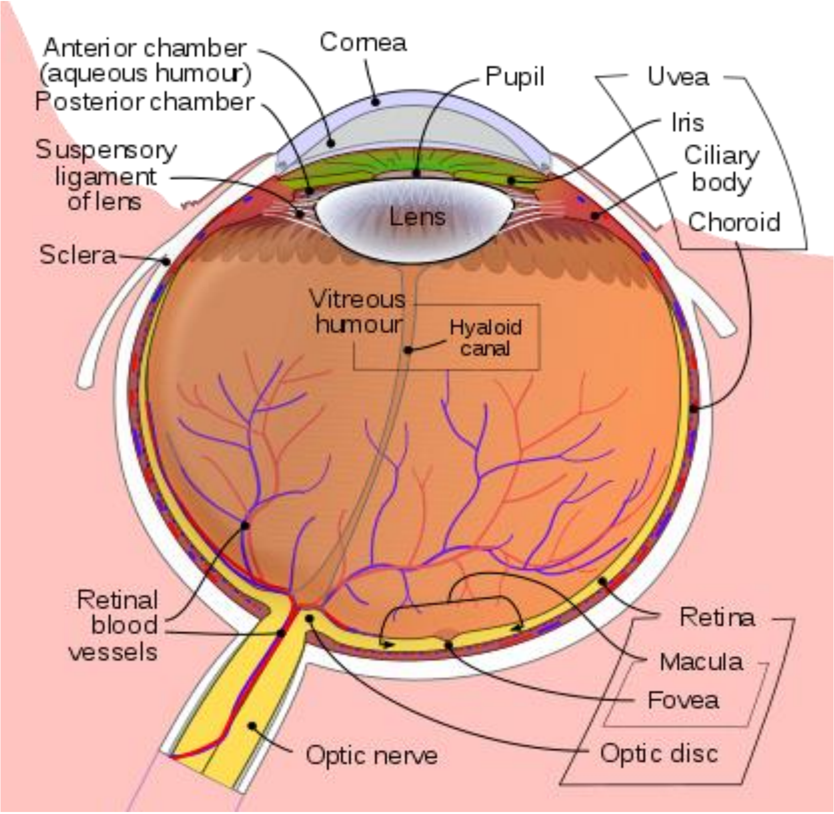
Dr. Esmaeil Najafi

MSc. Javad Khoramdel

دانشگاه
خواجه نصیرالدین طوسی
K. N. Toosi University
of Technology

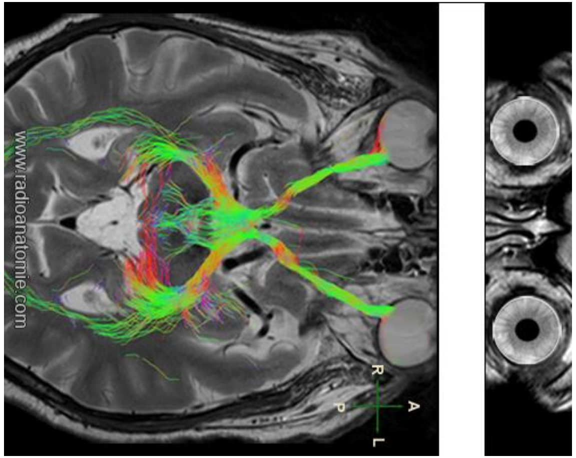
So how do human eyes work?

- Complex!
- Light passes through
  - Cornea, humours, lens refract light to focus
- Hit the retina
- Absorbed by photosensitive cells
- Info transmitted through optic nerve, processed by visual cortex



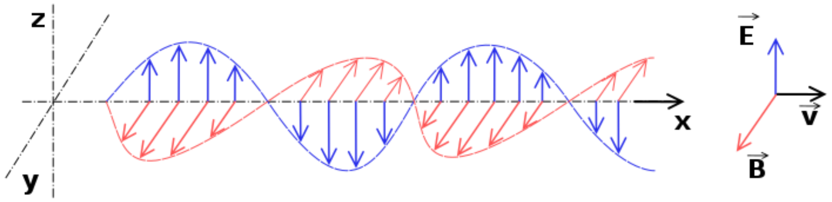
The brain and vision

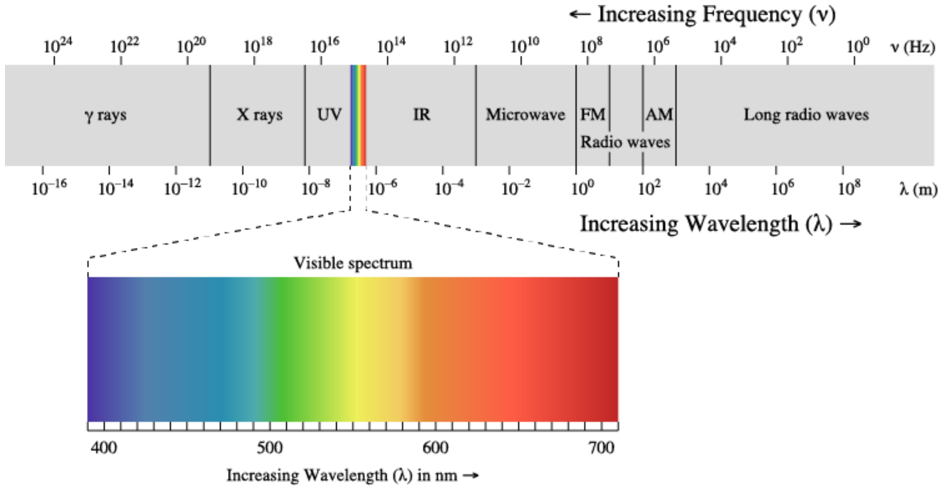
- Enormous processing power devoted to vision
- Visual cortex is largest “system” in the brain
  - 30% of the cerebral cortex
  - ⅔ of the electrical activity
- Lots of processing happening “subconsciously”



So what are we looking at anyway?

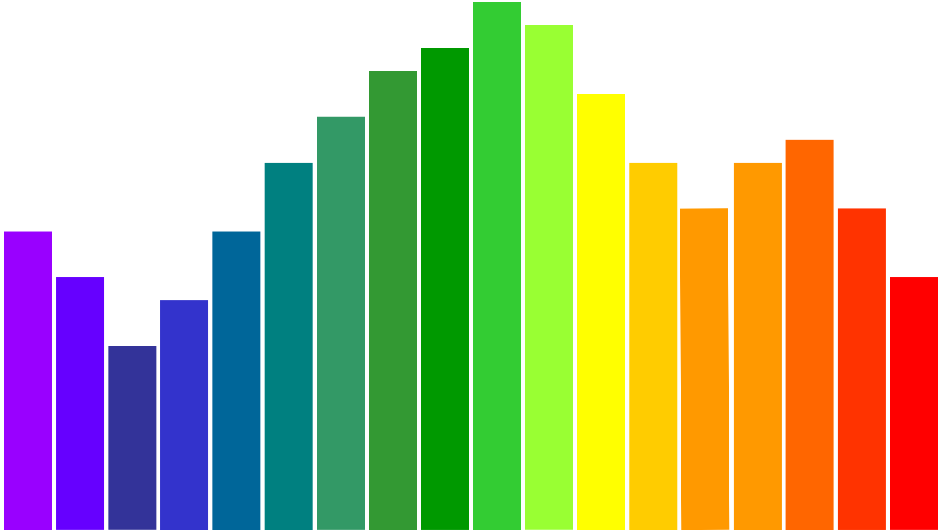
- Light is electromagnetic radiation.
- Visible light: ~400-700 nanometers.



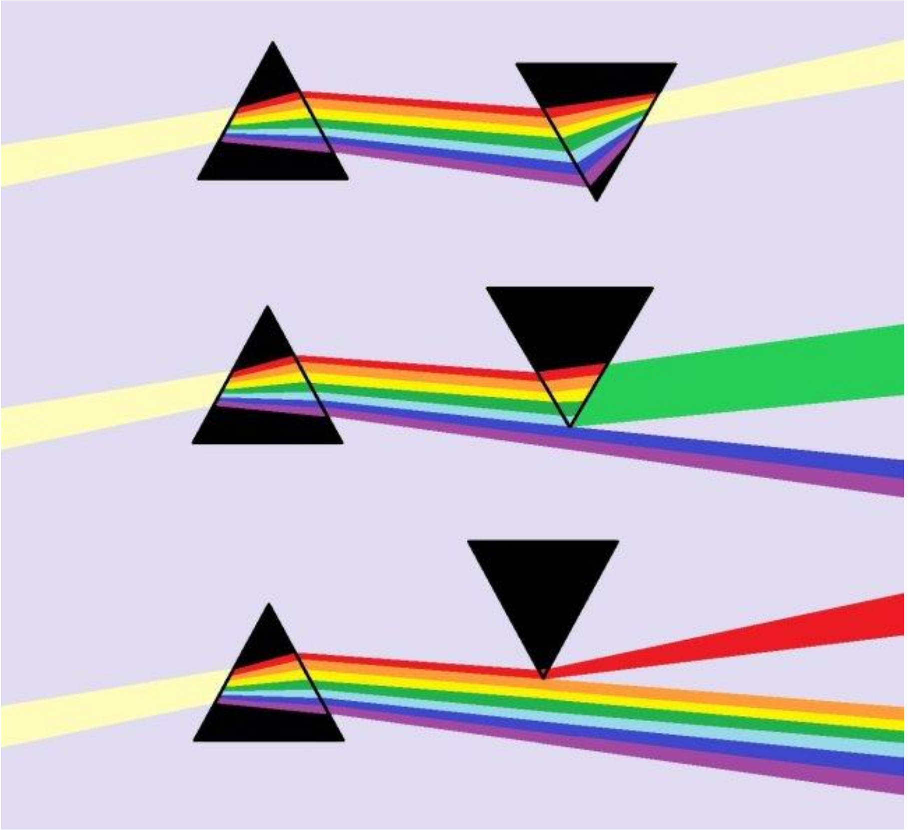


Light is a combination of waves

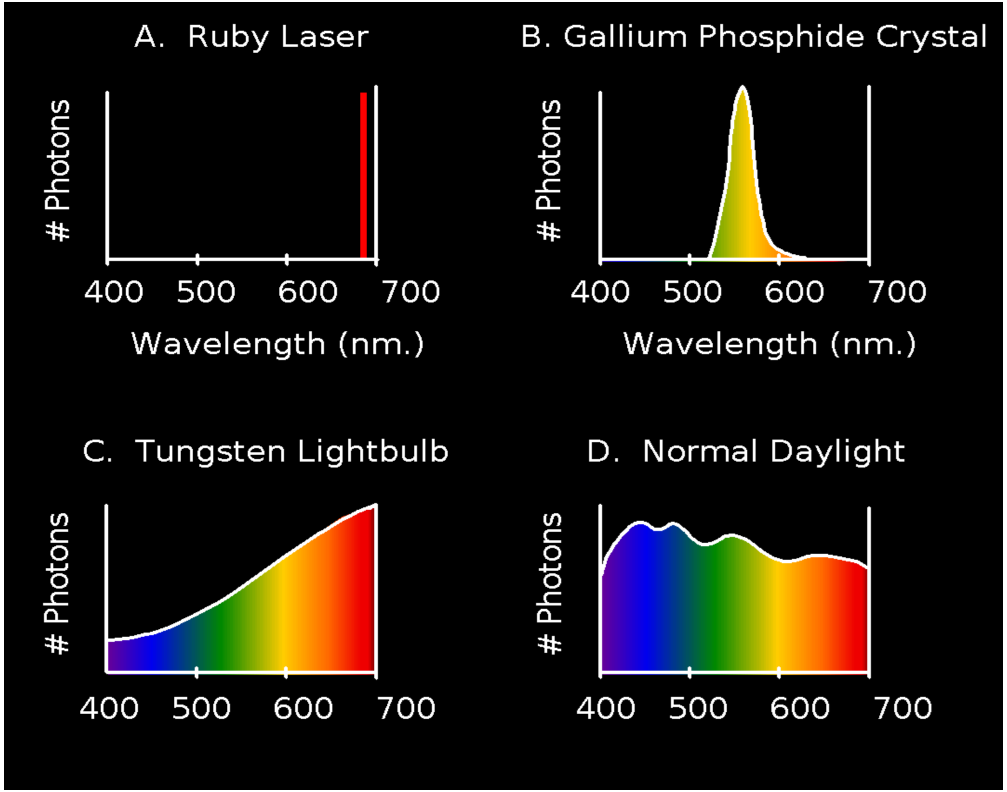
- A light ray can be described as a sum of its parts



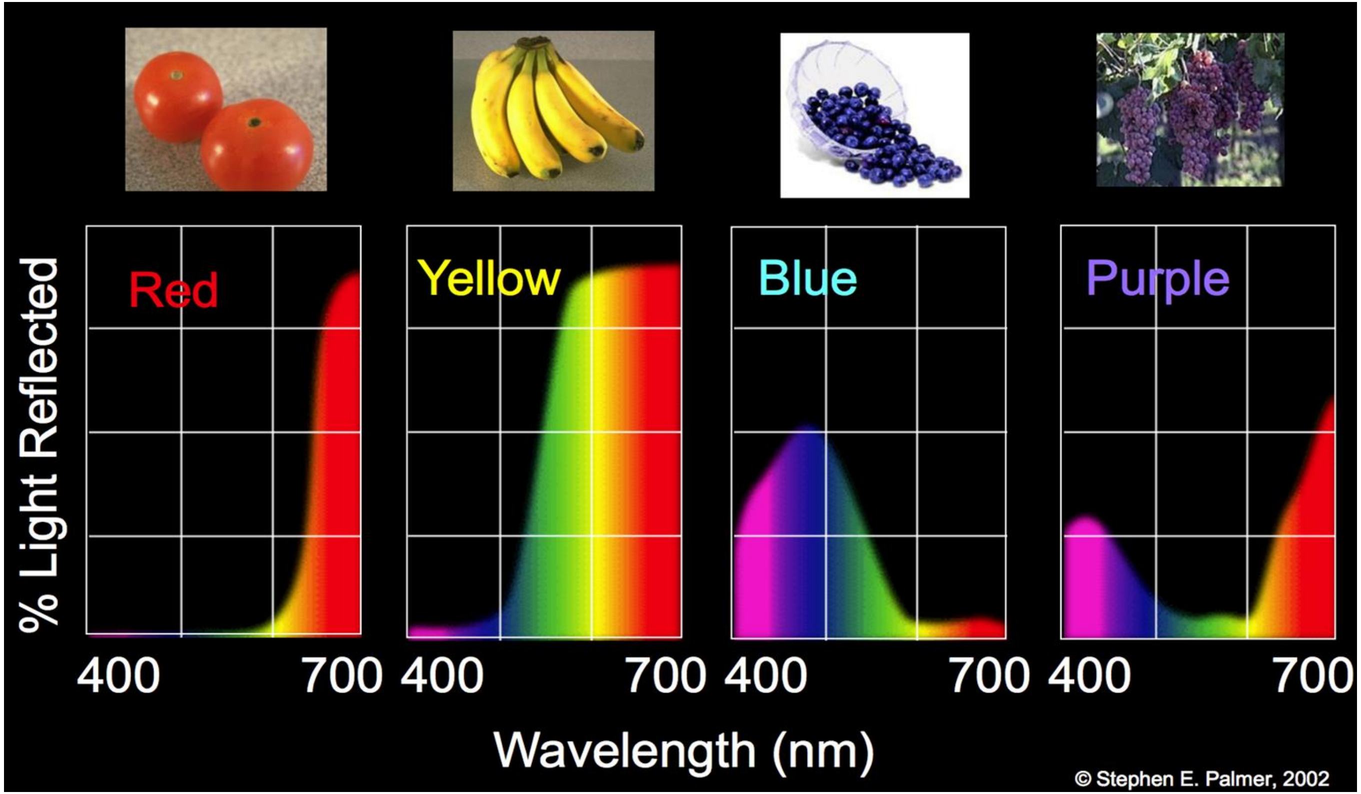
“White” light - all wavelengths



Sources of light are diverse!

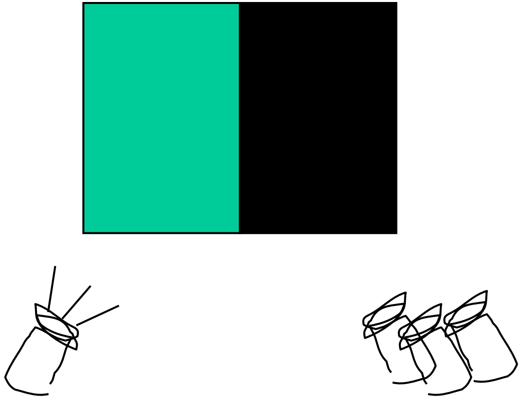


Objects reflect only some light

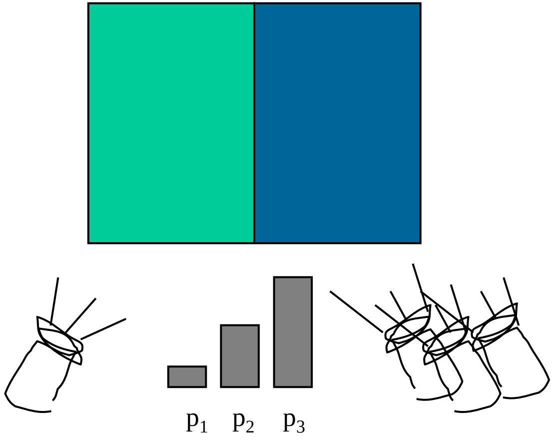


CIE 1931 and Color Matching

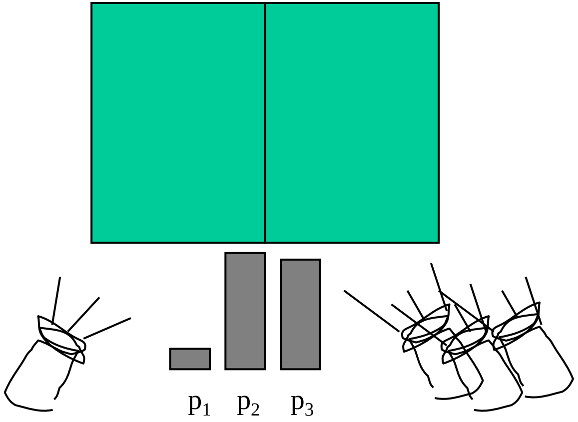
- Late 1920s William Wright and John Guild experimented with colors! (and people)
- Subjects get controls to 3 “primary” lights
- Show them a light
- Subject adjusts their lights to match the given light



No lights at the beginning, the plane is black



Starting with random amount of lights



The right combination is found! The right plane matches the template (left plane)

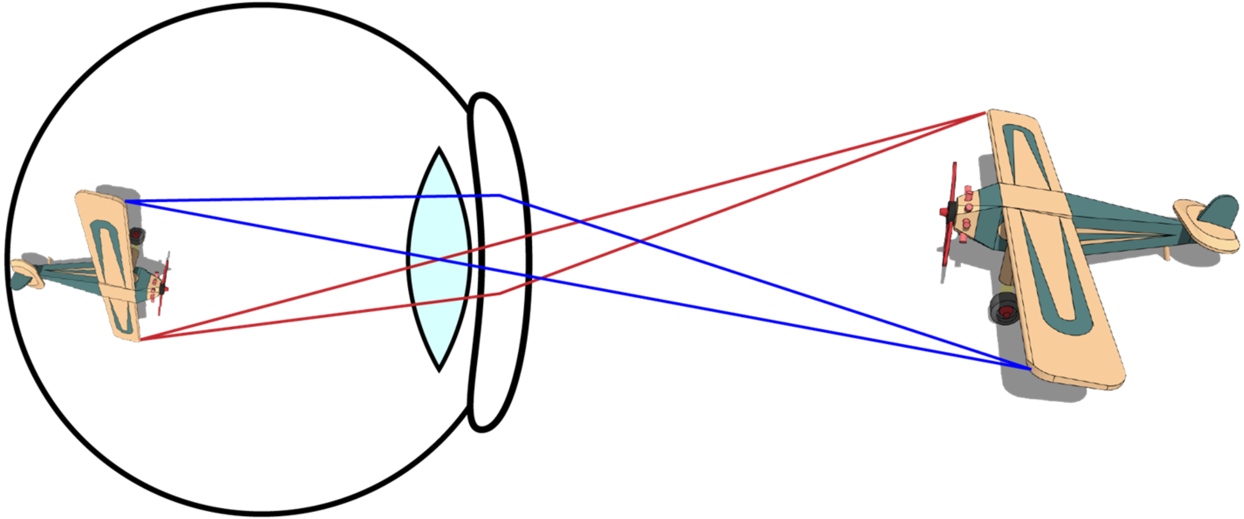
Results:

- Given 3 primaries people can match any color

- Colors seem to follow nice, linear rules: Grassman’s laws!
    - A=B+C => A+D=B+C+D
    - A=B+C => nA=nB+nC
    - A=B+C and D=B+C => A=D
- Light is combinations of individual wavelengths
    - If we can match any wavelength we can match any light
- Pick some primaries
    - Can mix those primaries to match any color.
- This method is made to trick humans, not be accurate

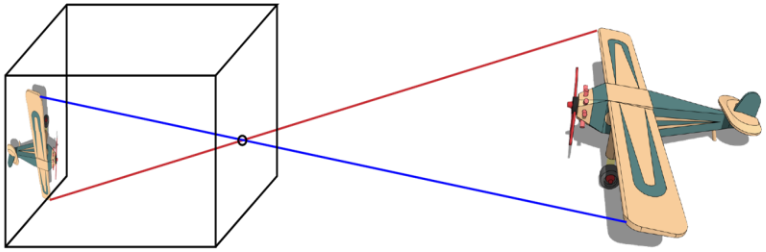
Too much information, now Let’s comeback to the human eye!

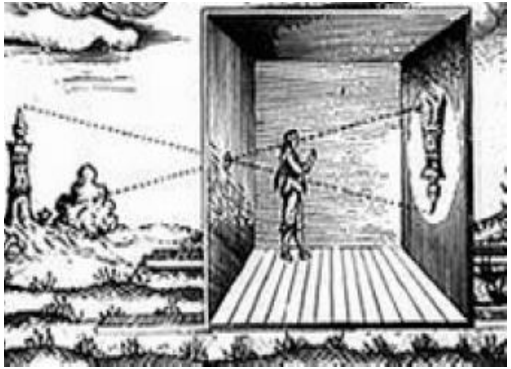
Eyes: projection onto retina



Model: pinhole camera

- A good lens can be modeled by a pinhole camera; i.e., each ray from the scene passes undeflected to the image plane
- Simple equations describe projection of a scene point onto the image plane ("perspective projection")





The pinhole camera ("camera obscura") was used by Renaissance painters to help them understand perspective projection.

Model: pinhole camera

- For convenience (to avoid an inverted image) we treat the image plane as if it were in front of the pinhole (i.e. the virtual image).

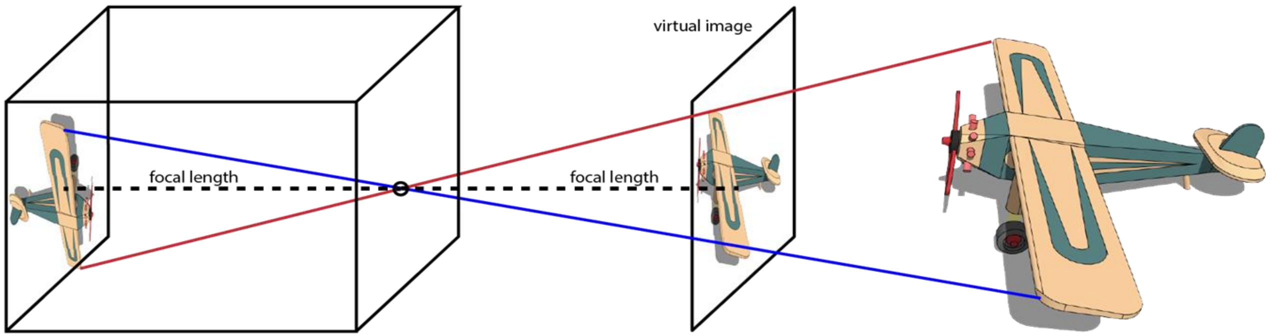
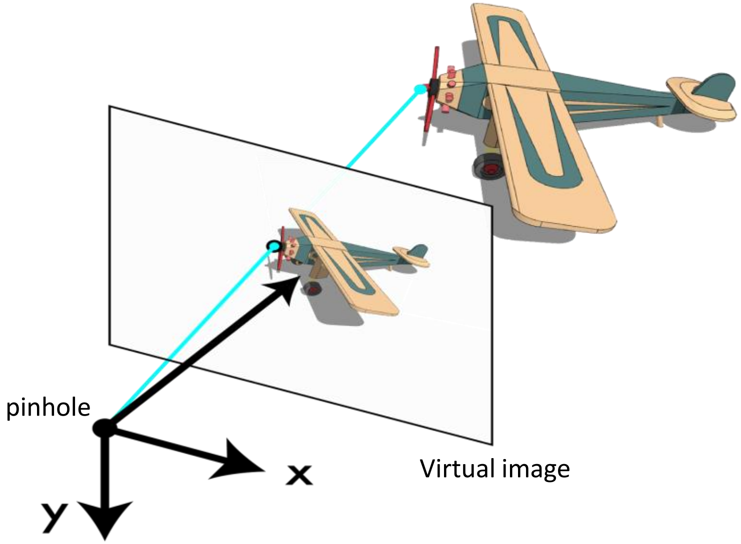
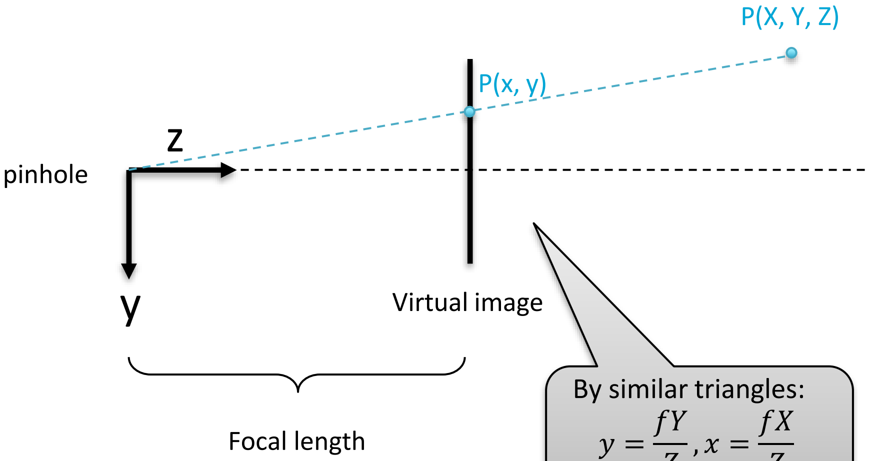


Image: 3d -> 2d projection of the world



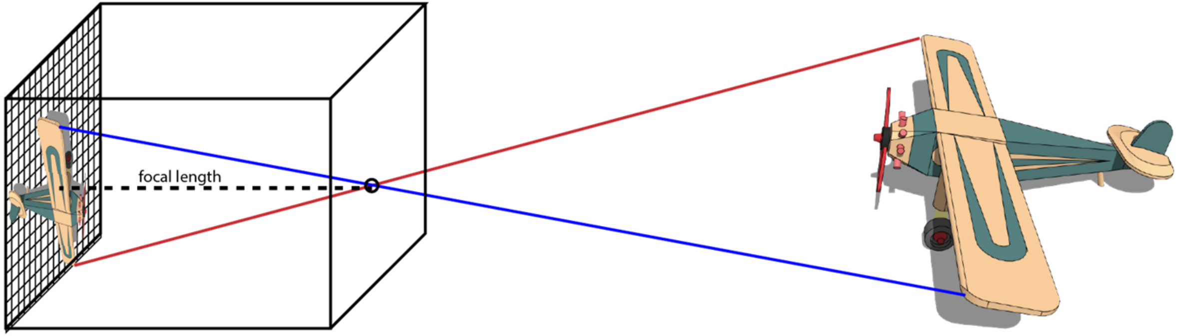


By similar triangles:

$$y = \frac{fY}{Z}, x = \frac{fX}{Z}$$

At each point we record incident light

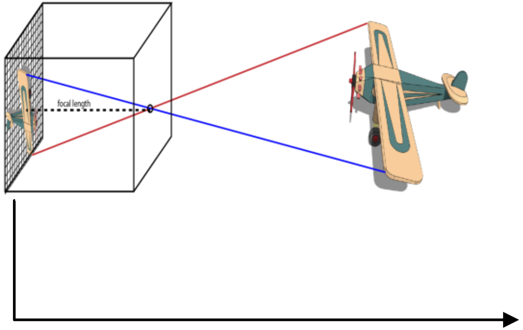
- An array of sensors is used to capture the light intensity at each location.

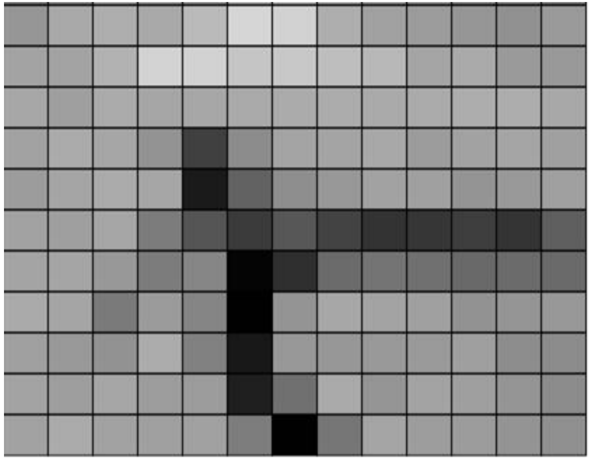


An image is a matrix of light

- An 8-bit ADC will convert the float values captured by each sensor.
- The values for an 8-bit varies between 0 and 255.
- The integer value for the brighter areas will be closer to 255.
- Each element of the matrix is called a “pixel”.

This image is called a “gray” image.





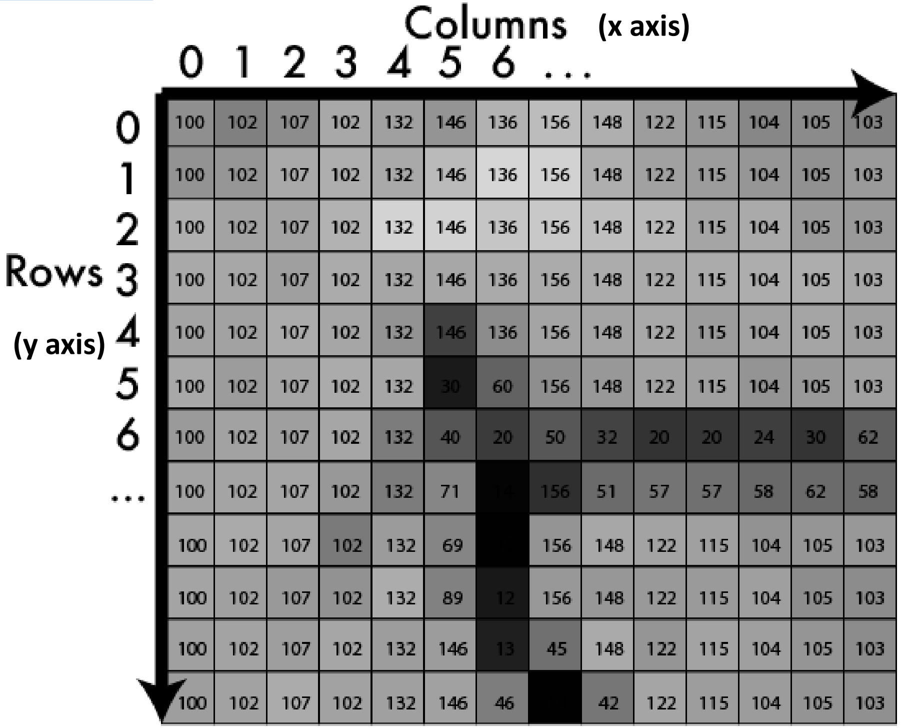
Light on the image plane

102	107	102	132	146	136	156	148	122	115	104	105	103
102	107	102	132	146	136	156	148	122	115	104	105	103
102	107	102	132	146	136	156	148	122	115	104	105	103
102	107	102	132	146	136	156	148	122	115	104	105	103
102	107	102	132	30	60	156	148	122	115	104	105	103
102	107	102	132	40	20	50	32	20	20	24	30	62
102	107	102	132	71	[illegible]	156	51	57	57	58	62	58
102	107	102	132	69	[illegible]	156	148	122	115	104	105	103
102	107	102	132	89	12	156	148	122	115	104	105	103
102	107	102	132	146	13	45	148	122	115	104	105	103
102	107	102	132	146	46	[illegible]	42	122	115	104	105	103

Light intensity captured by sensors

Addressing pixels

- Ways to index:
  - (x, y)
    - ✓ Like cartesian coordinates
    - ✓ (3,6) is column 3 row 6
  - (r, c)
    - ✓ Like matrix notation
    - ✓ (3,6) is row 3 column 6
- we use (x, y)
  - So does your homework!
  - Arbitrary
  - Only thing that matters is consistency



Binary image

- A binary image is a 2-D matrix which its elements are either 255 or 0.
- A gray image can be converted to a binary image with a threshold
  - If $intensity\ <\ 146$ set the value to zero.
  - If $intensity \geq 146$ set the value to 255.
- Note that there are more sophisticated ways to do the conversion (for example adaptive threshold) that we will cover later in this course

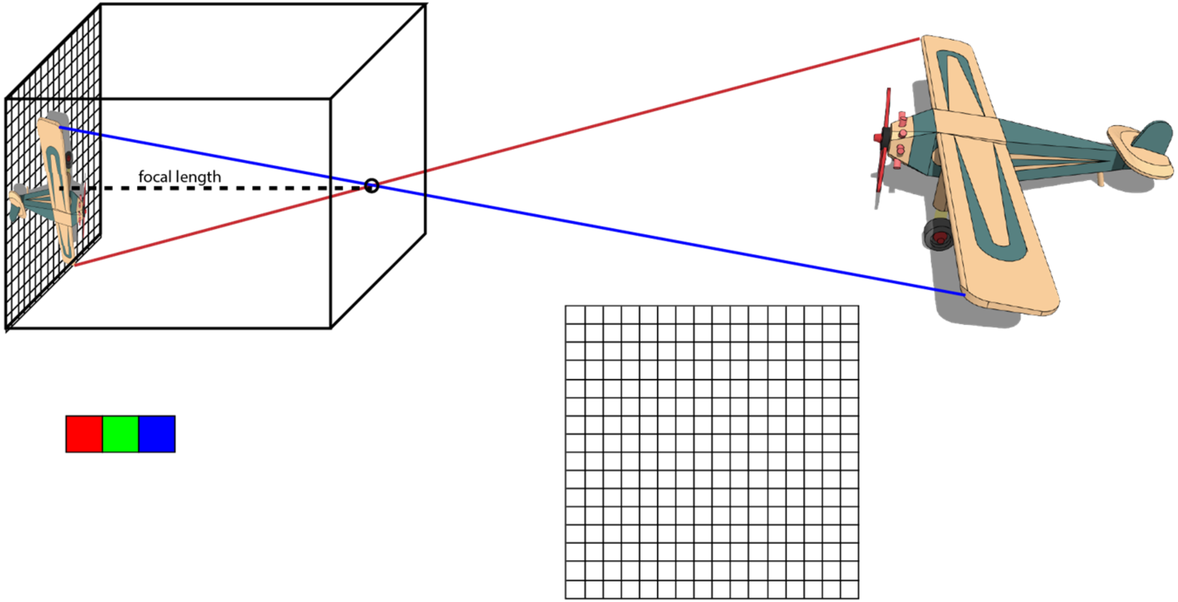
102	107	102	132	146	136	156	148	122	115	104	105	103
102	107	102	132	146	136	156	148	122	115	104	105	103
102	107	102	132	146	136	156	148	122	115	104	105	103
102	107	102	132	146	136	156	148	122	115	104	105	103
102	107	102	132	30	60	156	148	122	115	104	105	103
102	107	102	132	40	20	50	32	20	20	24	30	62
102	107	102	132	71		156	51	57	57	58	62	58
102	107	102	132	69		156	148	122	115	104	105	103
102	107	102	132	89	12	156	148	122	115	104	105	103
102	107	102	132	146	13	45	148	122	115	104	105	103
102	107	102	132	146	46		42	122	115	104	105	103

Gray image

255	255	255	255	255	255	255	255	255	255	255	255	255
255	255	255	255	255	255	255	255	255	255	255	255	255
255	255	255	255	255	255	255	255	255	255	255	255	255
255	255	255	255	255	0	255	255	255	255	255	255	255
255	255	255	255	0	0	255	255	255	255	255	255	255
255	255	255	0	0	0	0	0	0	0	0	0	0
255	255	255	255	255	255	0	0	0	0	0	0	0
255	255	255	255	255	0	0	255	255	255	255	255	255
255	255	255	255	255	0	0	255	255	255	255	255	255
255	255	255	255	255	0	0	255	255	255	255	255	255
255	255	255	255	255	0	0	255	255	255	255	255	255

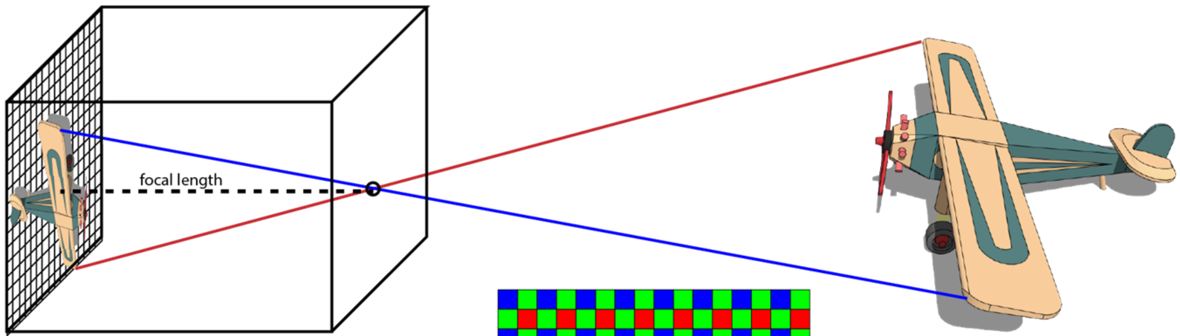
Binary image

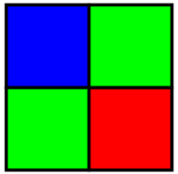
How do we record color?

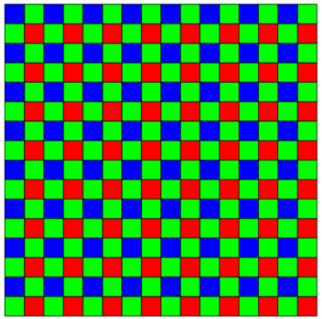


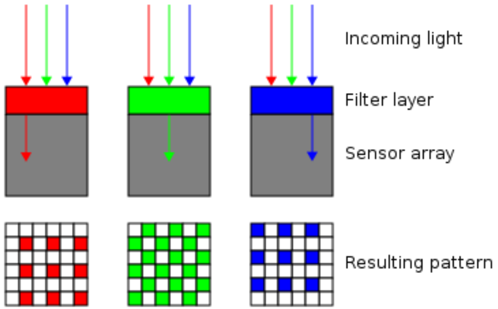
How do we record color?

- We have 3 principle colors.
- For each principle color, we can install a filter which only passes that color.



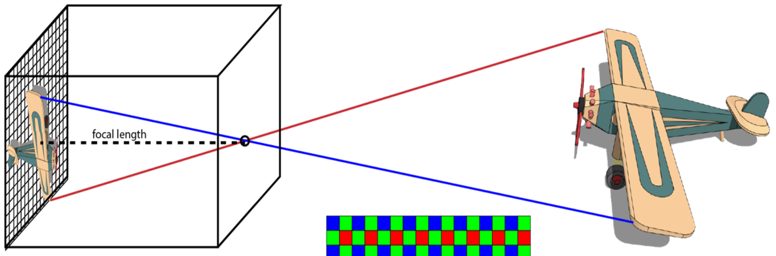


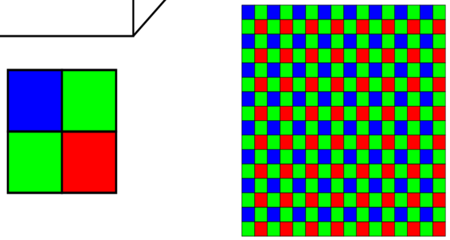


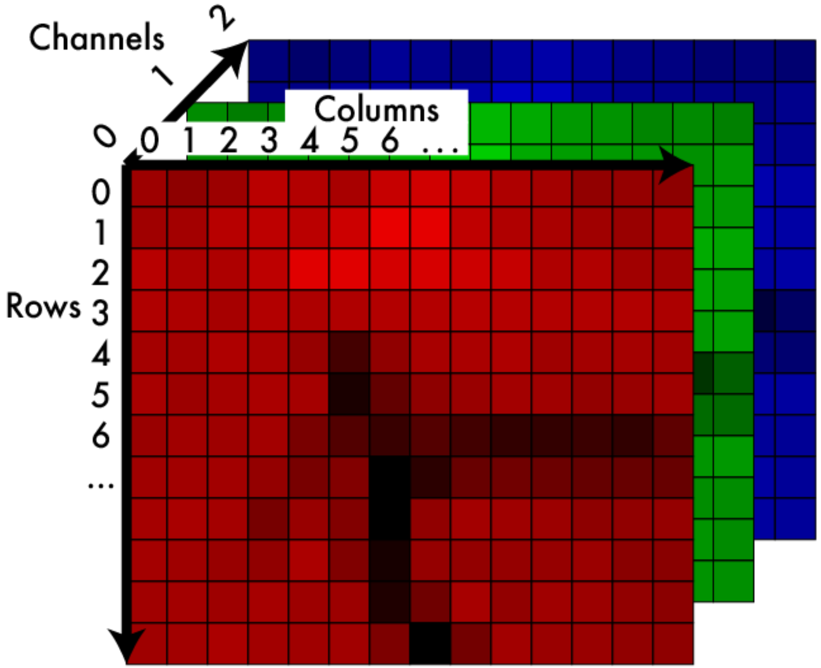


Color image: 3d tensor in colorspace

- For each location on the image plane, we have three values for the principle colors.

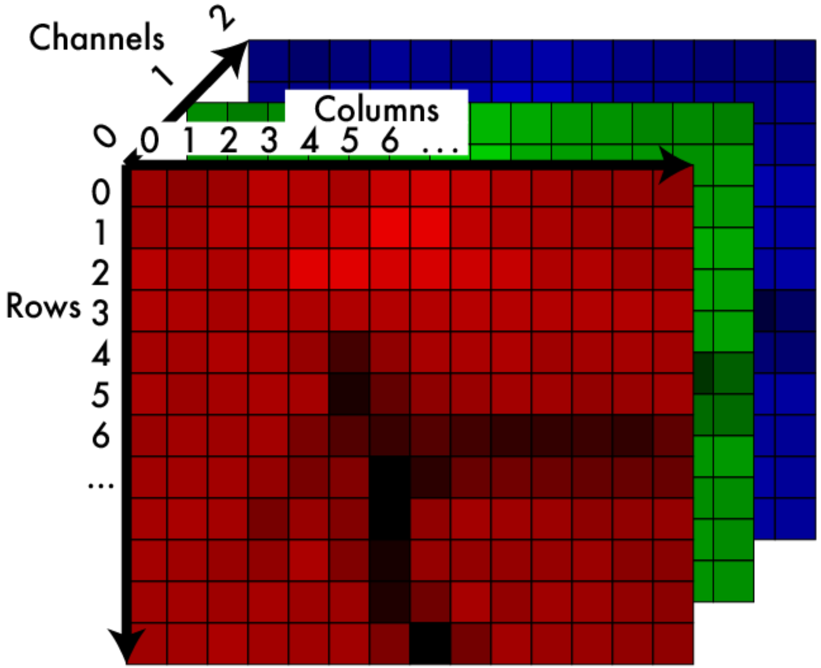






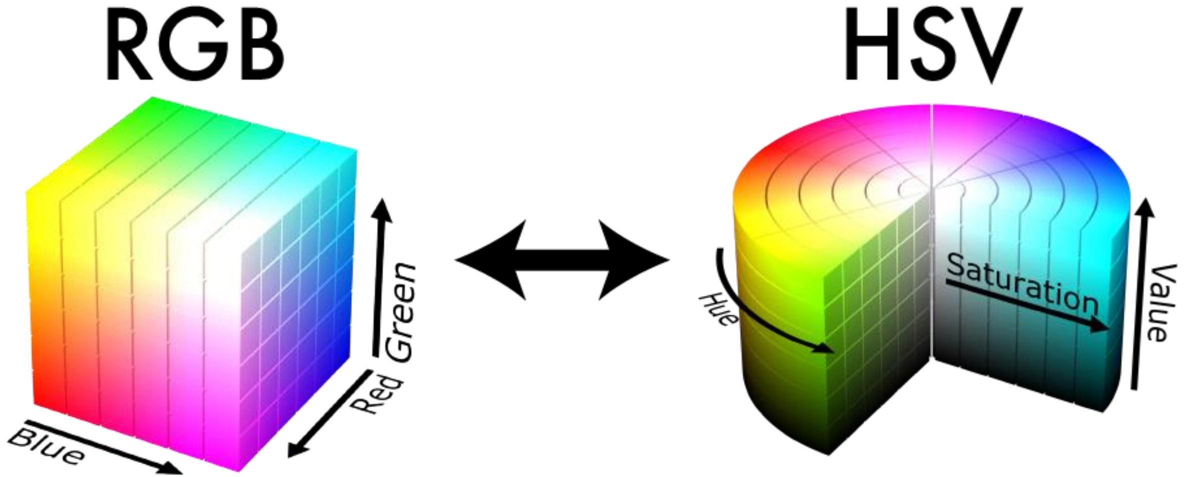
Addressing pixels in color image

- we use (x,y,c)
  - (1,2,0):
  - column 1, row 2, channel 0
- Still doesn’t matter, just be consistent
  - For example, there is also (c, x, y) representation

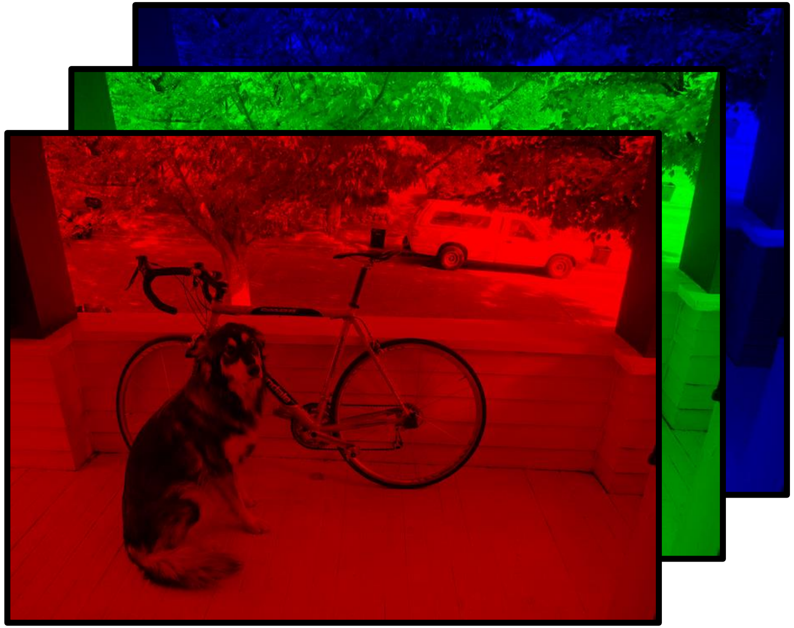


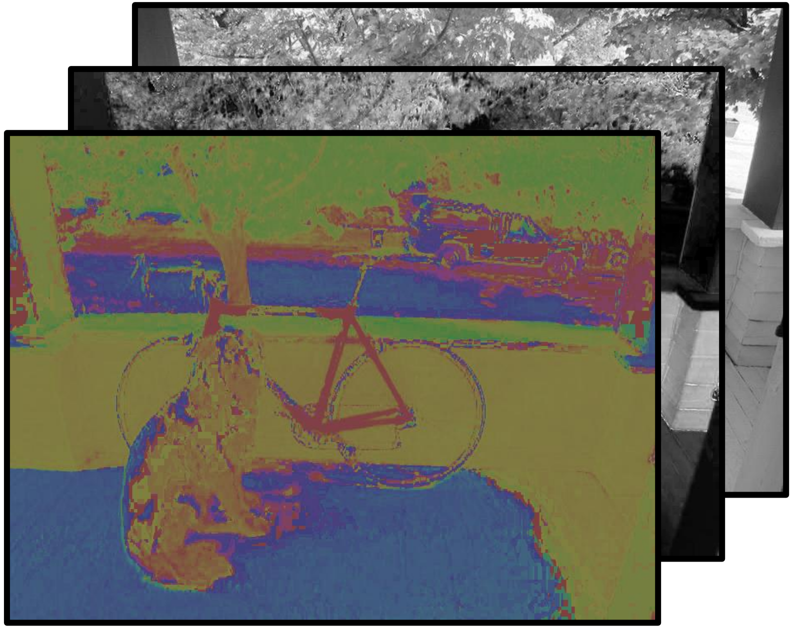
Other colorspaces are fun!

- The equations for converting RGB to HSV and vice versa are a little bit messy!
- You can easily find those if you are interested, but we won't cover that.
- We will use the built-in libraries in Python for the conversion.

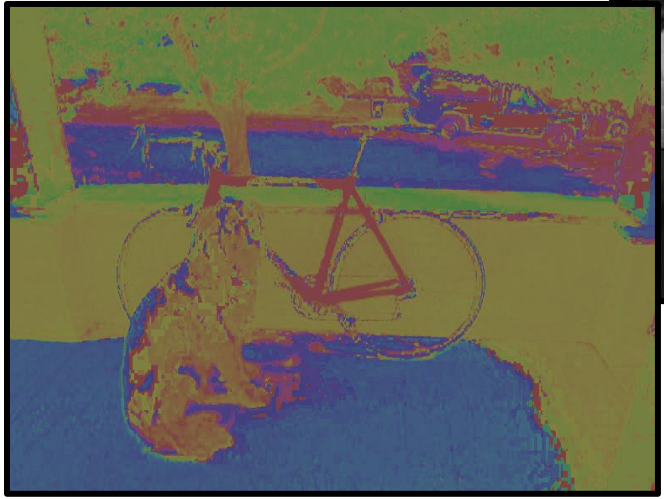


Still 3d tensor, different info

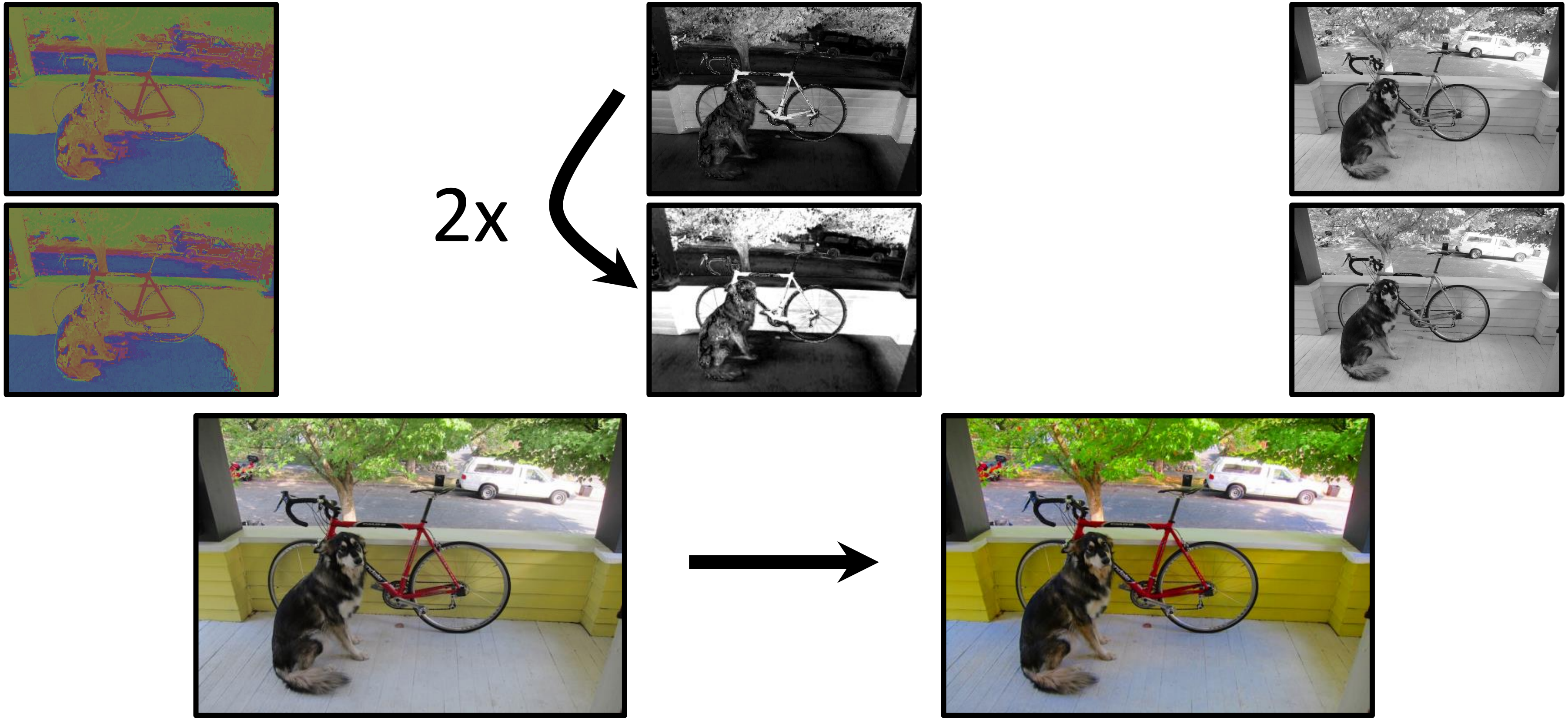




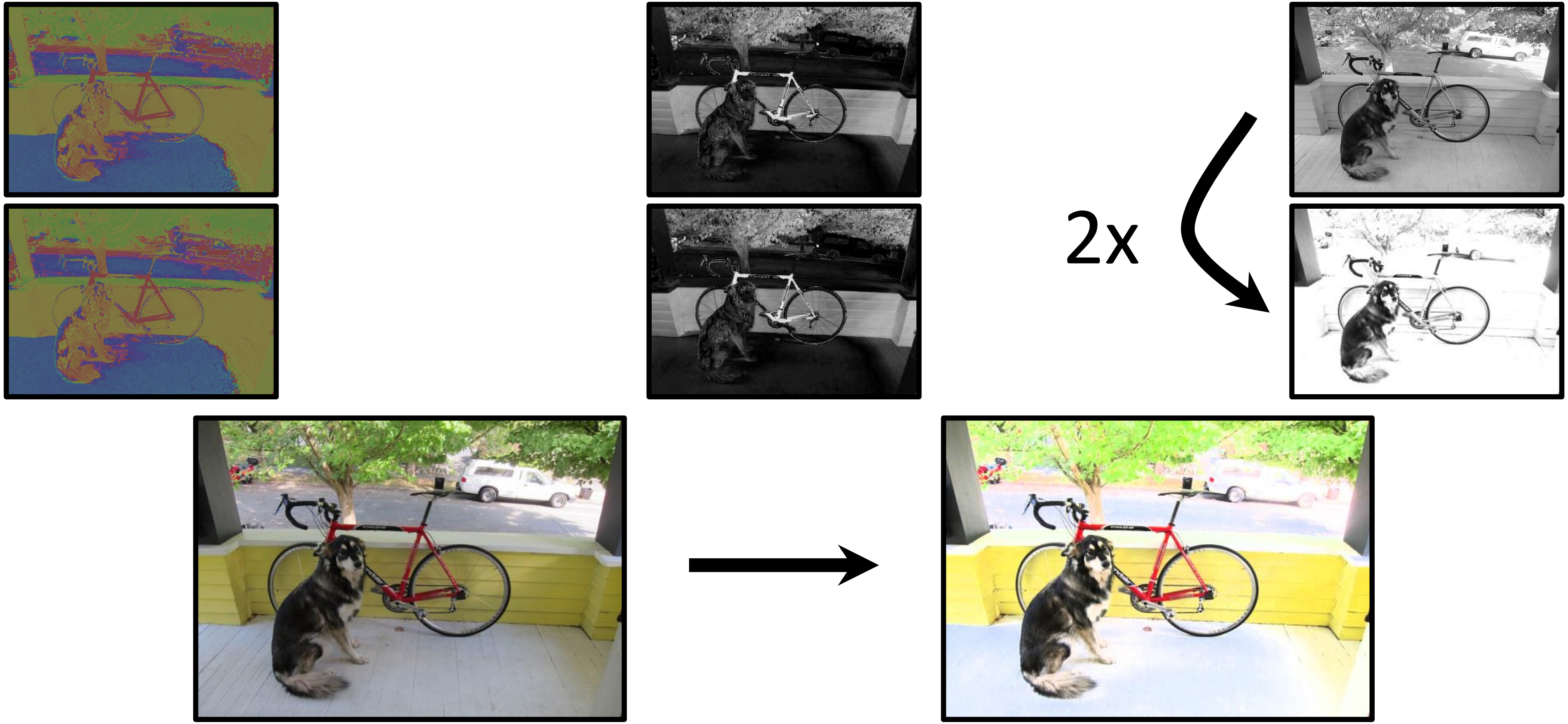
Hue, Saturation, Value



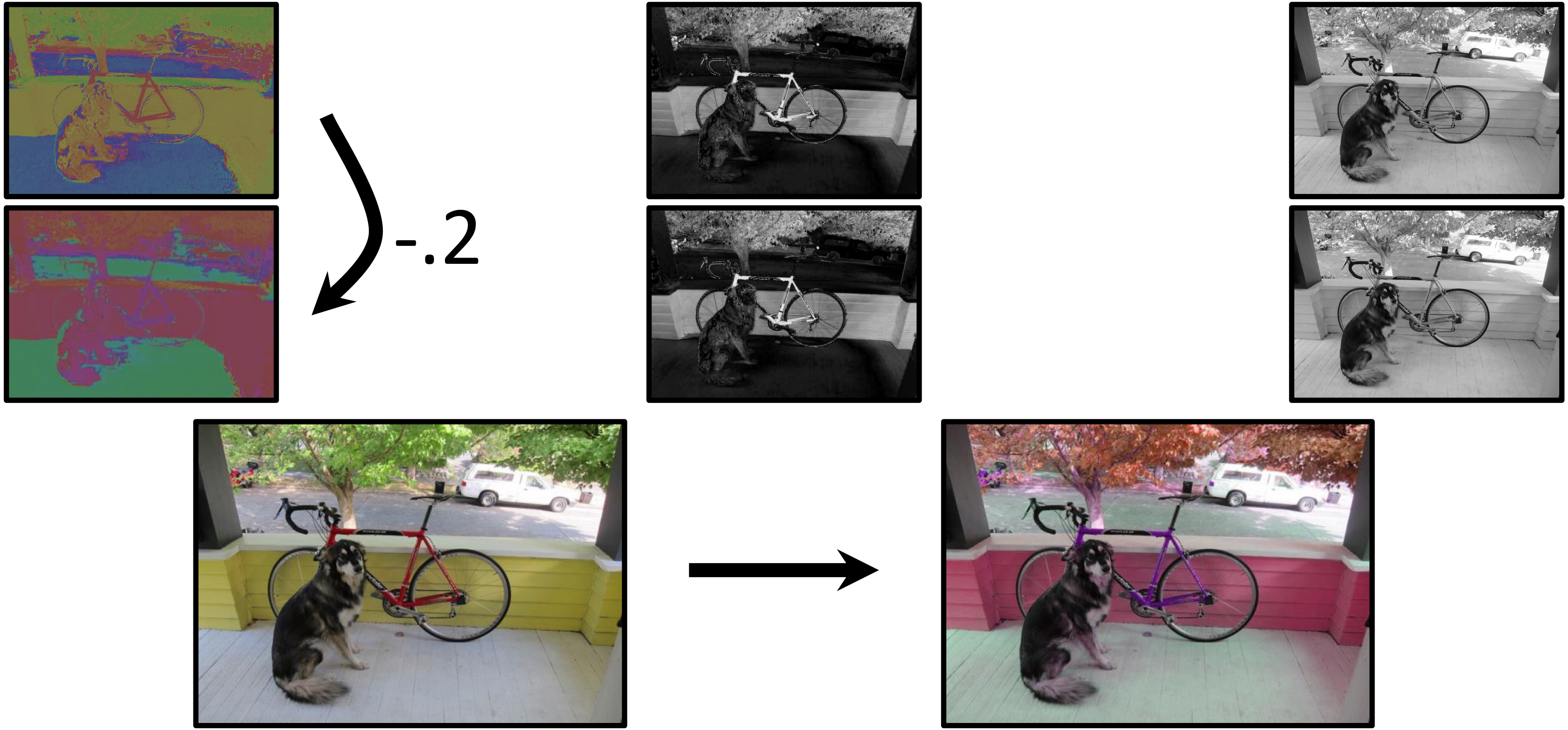
More saturation = intense colors



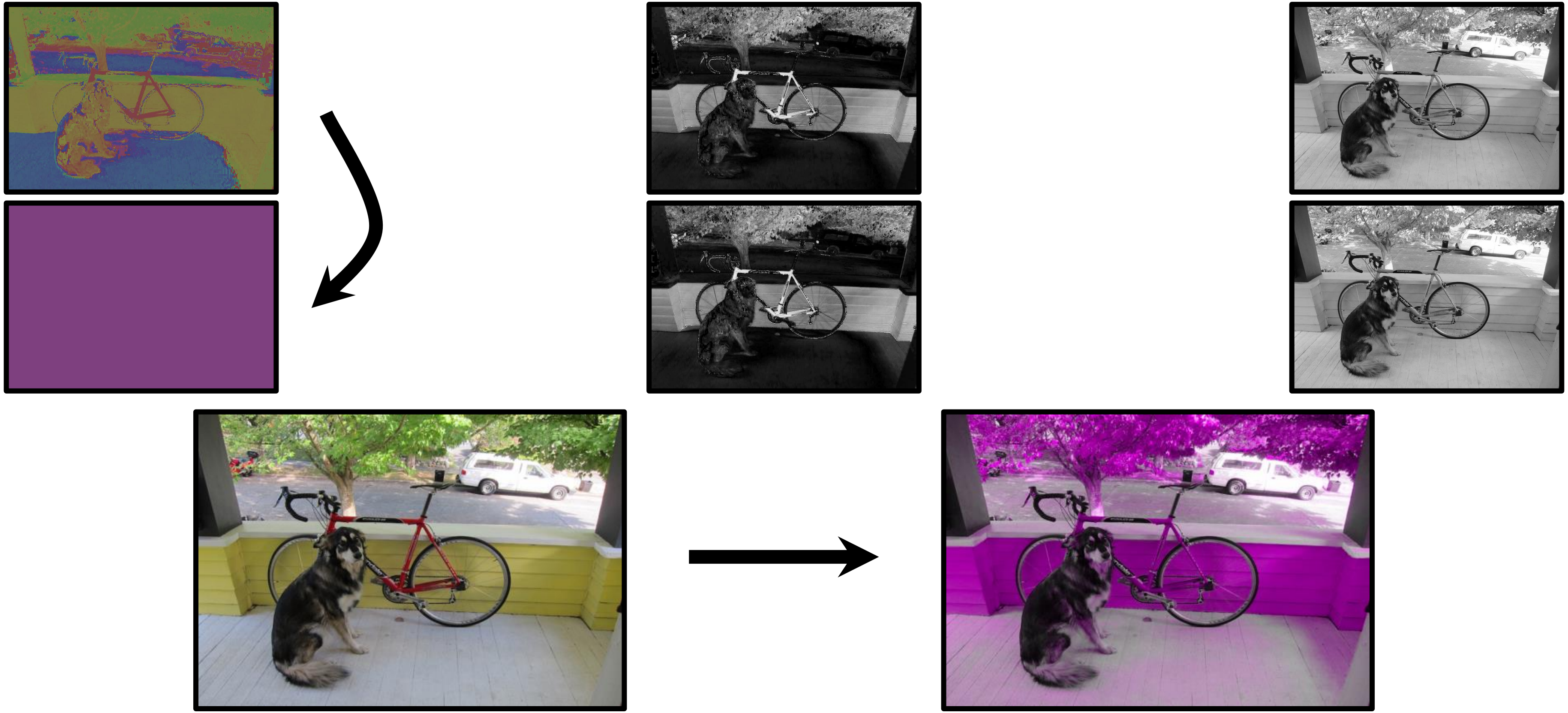
More value = lighter image



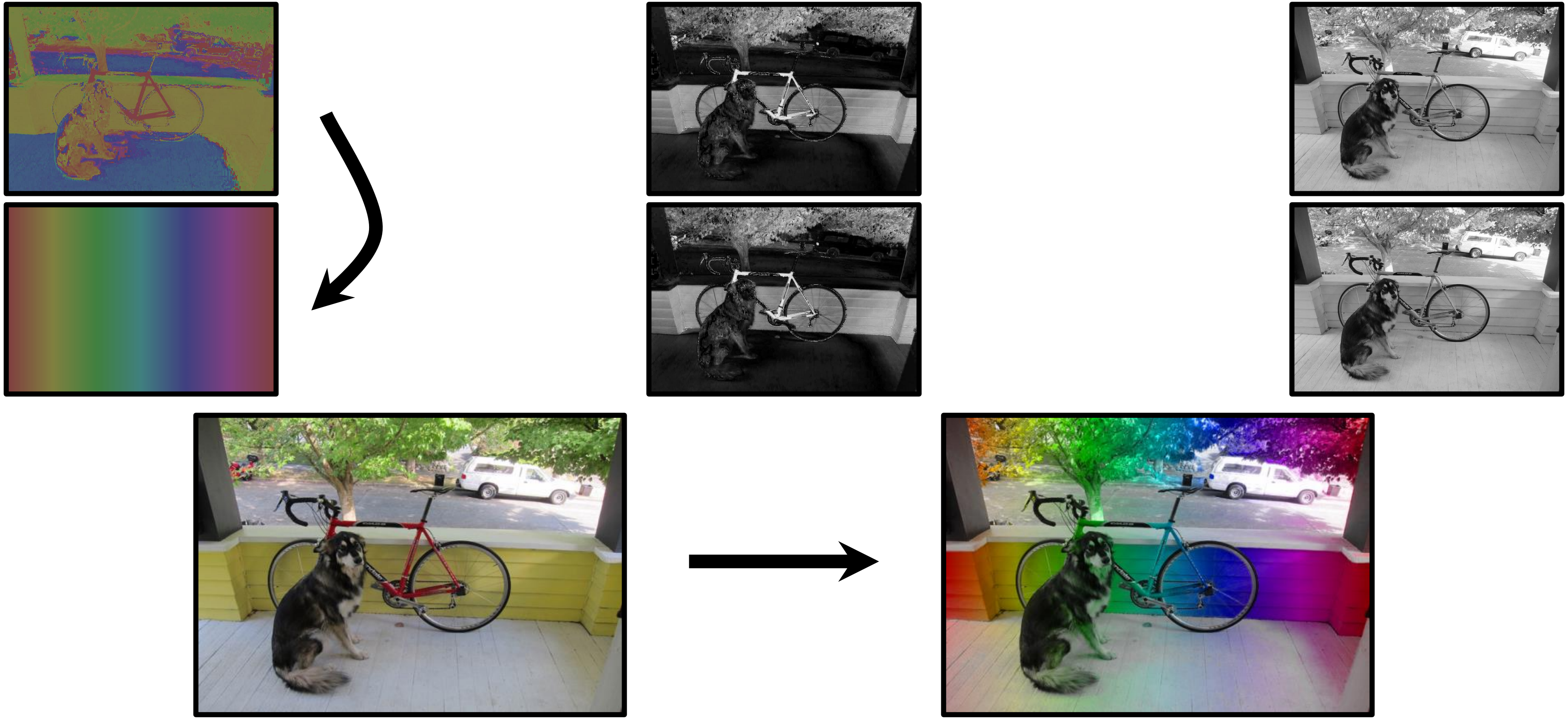
Shift hue = shift colors



Set hue to your favorite color!



Or pattern...



Just one more thing to know: Digital cameras

- CCD type (Charge coupled device)
  - Charge accumulates during exposure
  - Charges are transferred out to shift registers, digitized and read out sequentially
- CMOS type (complementary metal oxide on silicon)
  - Light affects the conductivity (or gain) of each photodetector
  - Digitized and read out using a multiplexing scheme
- Main design factors
  - Number and size of sensor elements
  - Chip size
  - ADC resolution

CCD cameras

CMOS cameras