



January 5, 2005

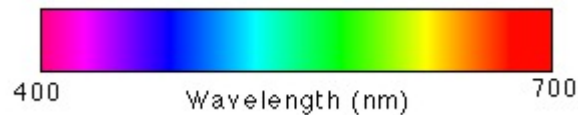


Agenda

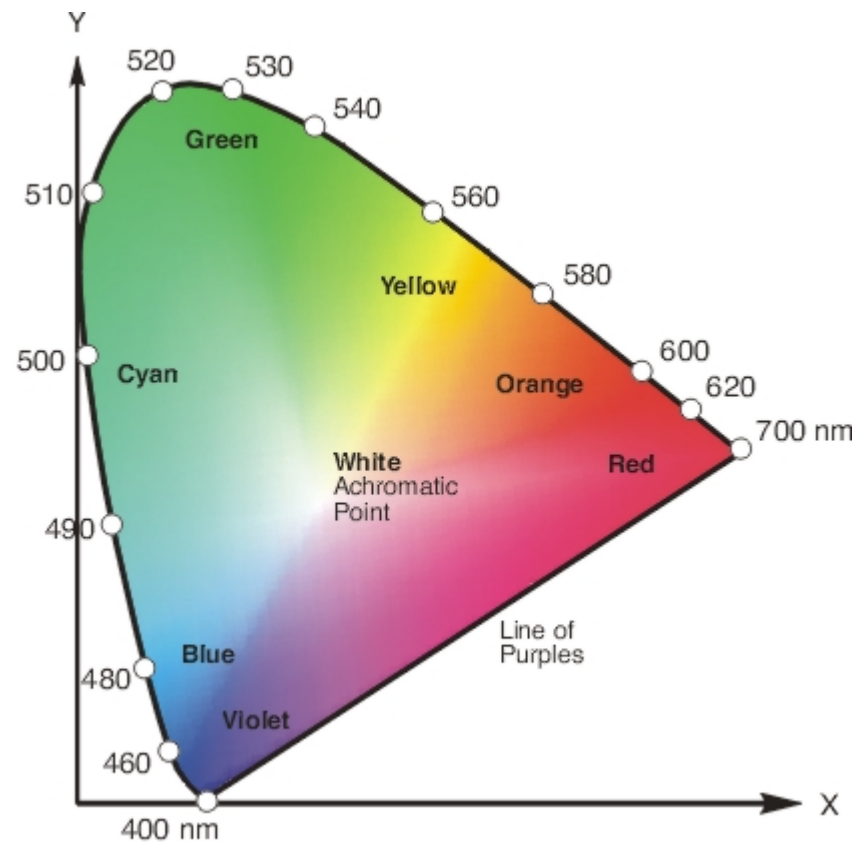
- Color and colorspaces
- Numbers and Java
- Feature detection

What are Colors?

- Frequencies are one dimensional...

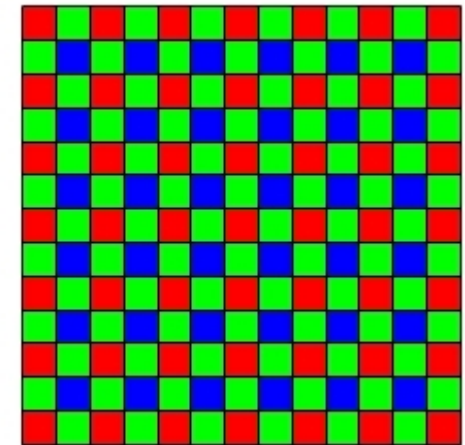


- But human perception of color is not!



Humans and Vision

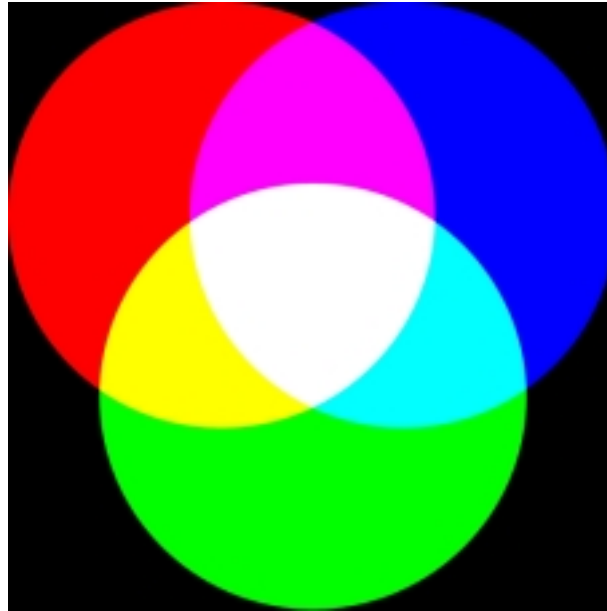
- We use cones to detect red, green, and blue
- So computer monitors use the same, with one byte per channel (RGB).
 - image 640x480 = 900 KB !
 - computers can cheat...
- ...like our cameras:
interpolate pixel values



Bayer filter

Colorspaces

- RGB good for light

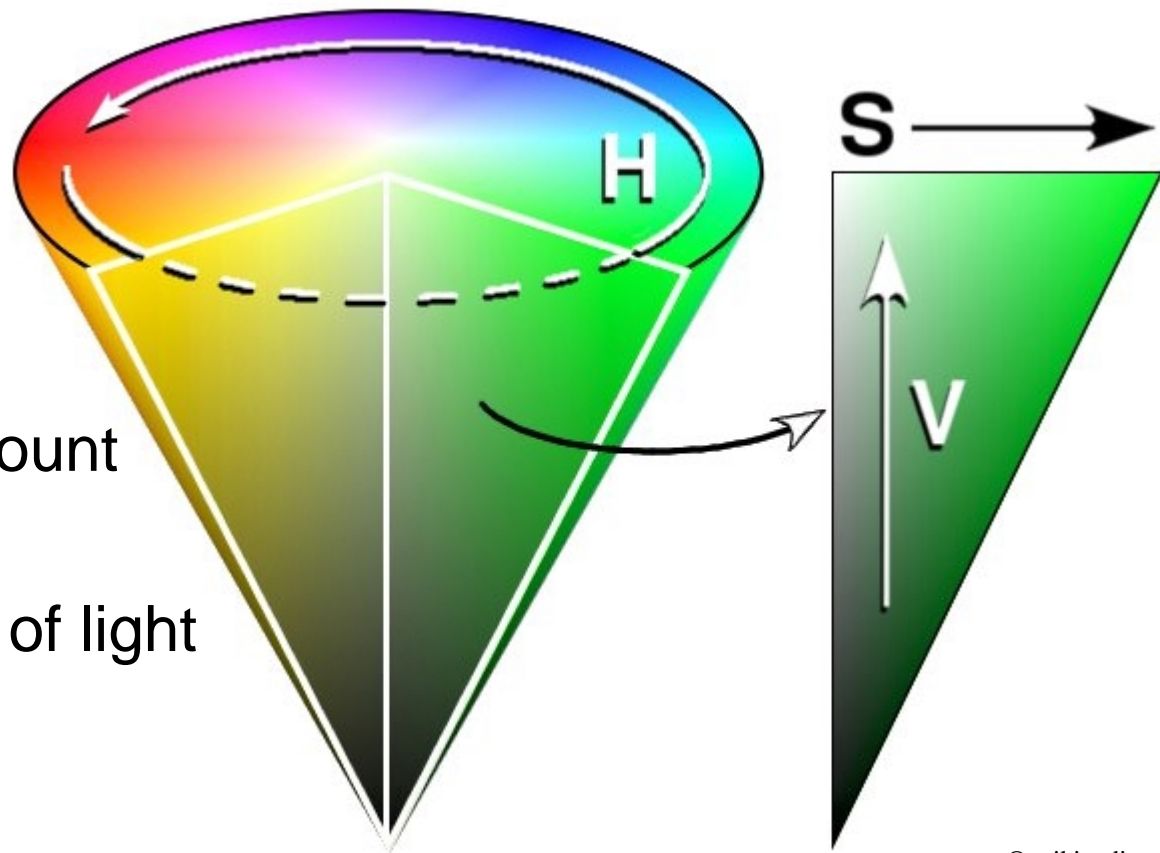


wikipedia

- CYMK good for pigment
 - ... but both mix color, tint, and brightness

Maslab Colorspace: HSV

- Hue (color):
- Saturation (amount of color)
- Value (amount of light and dark)





Using the colorspace

- We provide the code to convert to HSV
- For hue: 360 degrees mapped to 0 to 255
- Red is both 0 and 255!
- White is low saturation, but can have any hue.
- Black is high value, but can have any hue.



Tips on Differentiating Colors

- Globally define thresholds
- Self-calibrate for different lights
- Use the gimp/bot client on real images



How HSV values are stored

- Uses Hexadecimal (base 16)
 - 1,2,3,4,5,6,7,8,9,A,B,C,D,E,F,10,11,12...
 - 0x12 = 18
- A color is four bytes = 8 hexadecimal numbers
 - Alpha
 - Hue
 - Saturation
 - Value



Manipulating HSV values

- Use masks to pick out parts:
 - $0x12345678 \& 0x00FF0000 = 0x00340000$
- Shift to move parts around:
 - $0x12345678 \gg 8 = 0x00123456$
- Example: $\text{hue} = (X \gg 16) \& 0xFF$



A note on java...

- All java types are signed
 - A byte ranges from -128 to 127
 - Coded in two's complement: to change sign, flip every bit and add one
- Don't forget higher order bits
 - `(int) 0x0000FF00 = (int) 0xFF00`
 - `(int) ((byte) 0xFF) = (int) 0xFFFFFFFF`
- Watch out for shifts
 - `0xFD000000 >> 8 = 0xFFFD0000`



Example

- How about

```
int v = image.getPixel(25,25); // v = 0x8AC12390
byte hue = (v >> 16) & 0xFF //hue = 0xC1
if (hue > 200)
    foundRedBall();
```



Solution

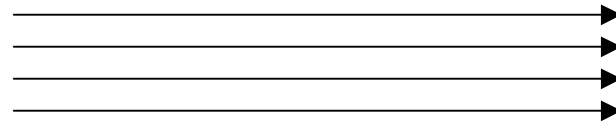
- Use

```
int v = image.getPixel(25,25); // v = 0x8AC12390
int hue = (v >> 16) & 0xFF //hue = 0xC1
if (hue > 200)
    foundRedBall();
```



Performance...

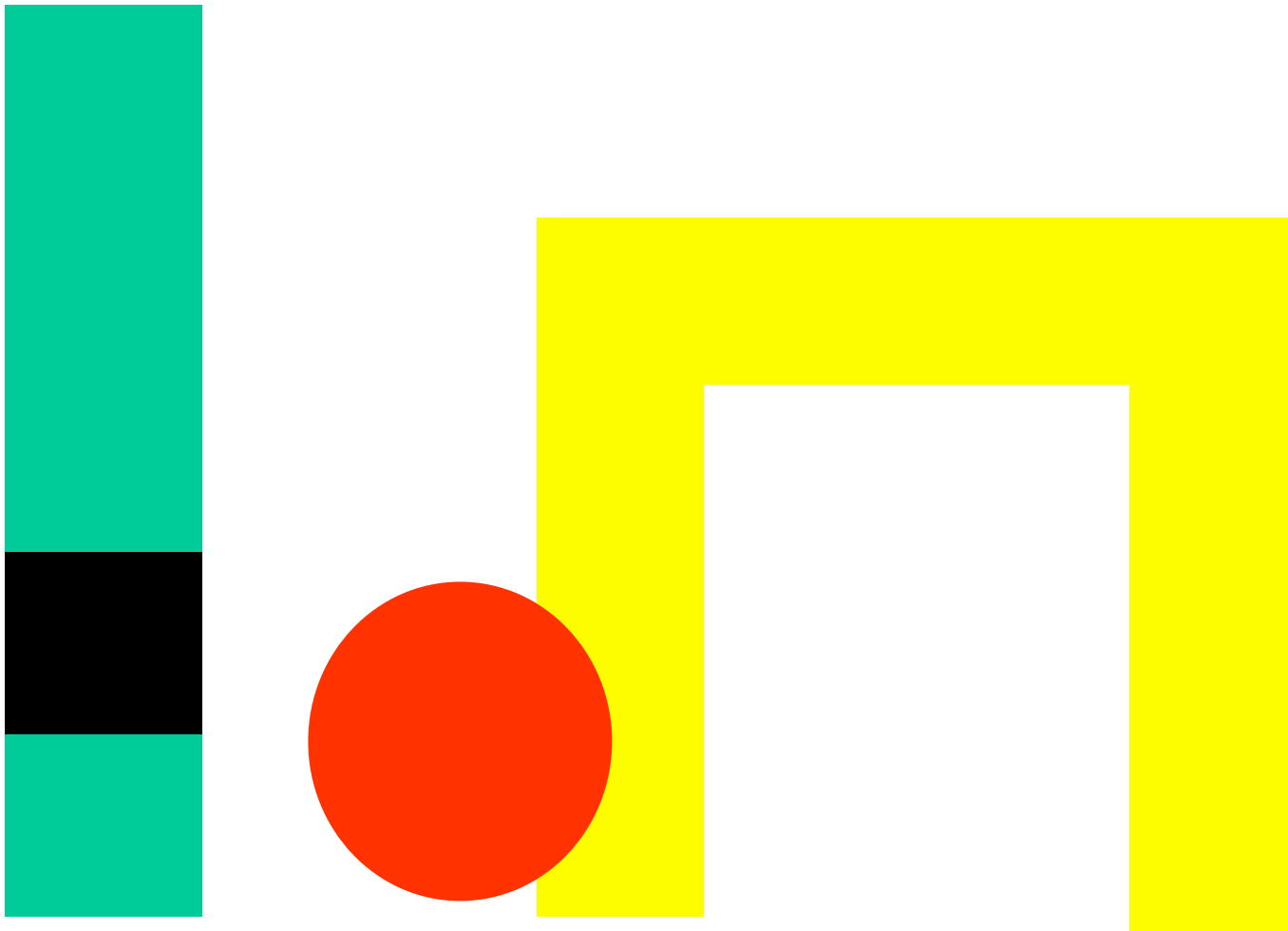
- Getting an image performs a copy
 - `Int[] = bufferedImage.getRGB(...)`
- Getting a pixel performs a multiplication
 - `int v = bufferedImage.RGB(x,y)`
 - `offset = y*width + x`
- Memory in rows, not columns...so go across rows and then down columns





Feature Detection...

and other Concepts





Maslab Features

- Red balls
- Yellow Goals
- Blue line
- Blue ticks
- Bar codes

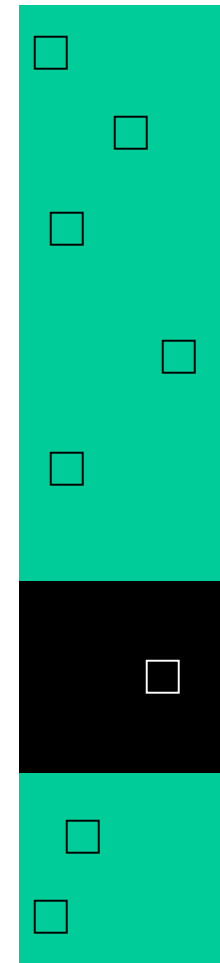
Blue line ideas

- Search for 'n' wall-blue pixels in a column
- Make sure there's wall-white below?
- Candidate voting
 - in each column, list places where you think line might be
 - find shortest left to right path through candidates



Bar code ideas

- Look for green and black
- Is there not-white under the blue line?
- Check along a column to determine colors
- RANdOm SAmpLe Consensus (RANSAC)
 - Pick random pixels within bar code
 - Are they black or green?



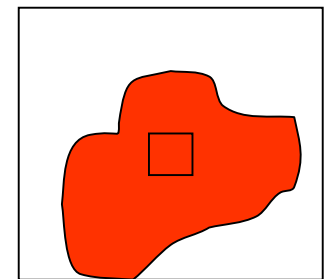
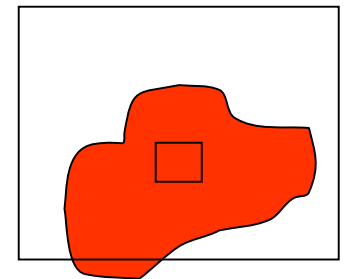
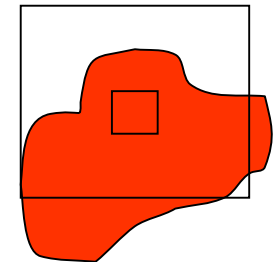
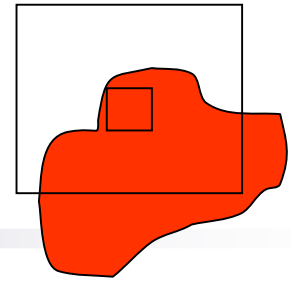


Finding a single color object

- Matched filter: convolve the image with a matched filter
 - computationally expensive
 - don't know the scale

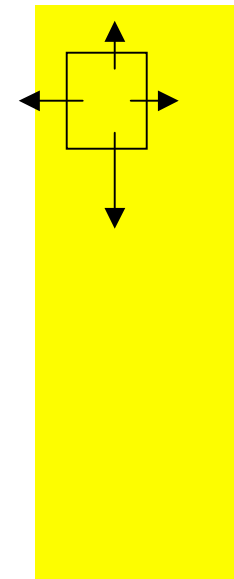
Other things to try

- Look for a red patch
- Set center to current coordinates
- Loop:
 - Find the new center based on pixels within d of the old center
 - Enlarge d and recompute
 - Stop when increasing d doesn't add enough red pixels



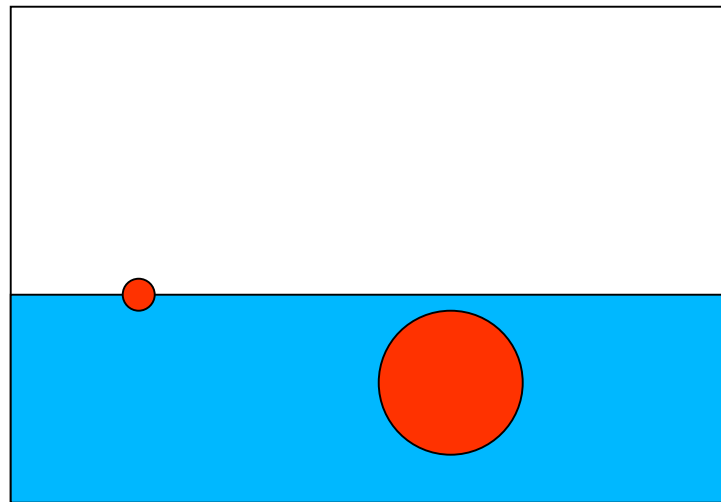
Or try fitting a rectangle

- Scan image for a yellow patch
- In each direction, loop:
 - Make rectangle bigger
 - If it doesn't add enough new yellow pixels, then stop



Estimating distance

- Closer objects are bigger
- Closer objects are lower





Feature-based processing

- Image processing for navigation
- In each frame, list 'corners' – such as the blue tick marks
- Match corners from one image to the next
- Estimate the rigid 3D transformations to that best map the corners



Reminders

- Basics to get you started
 - (cool advanced stuff on Monday)
- Try out your own algorithms! Have fun!
- Must prune out silly solutions:
 - Noise
 - Occlusion
 - Acute viewing angles
 - Overly large thresholds



Updates on Rules

- Robot must fit in tub
- There will be yellow field goal posts over the goals (above the yellow line)
- Using outside parts: cost = how much it would cost another team to have similar functionality
- Also, don't forget to refresh wiki periodically during the day and check for updates



Your job for today

- Finish yesterday's activity
- Read a barcode
- Work on Friday's check point