

The Vision Pipeline and Color Image Segmentation

15-494 Cognitive Robotics
David S. Touretzky &
Ethan Tira-Thompson

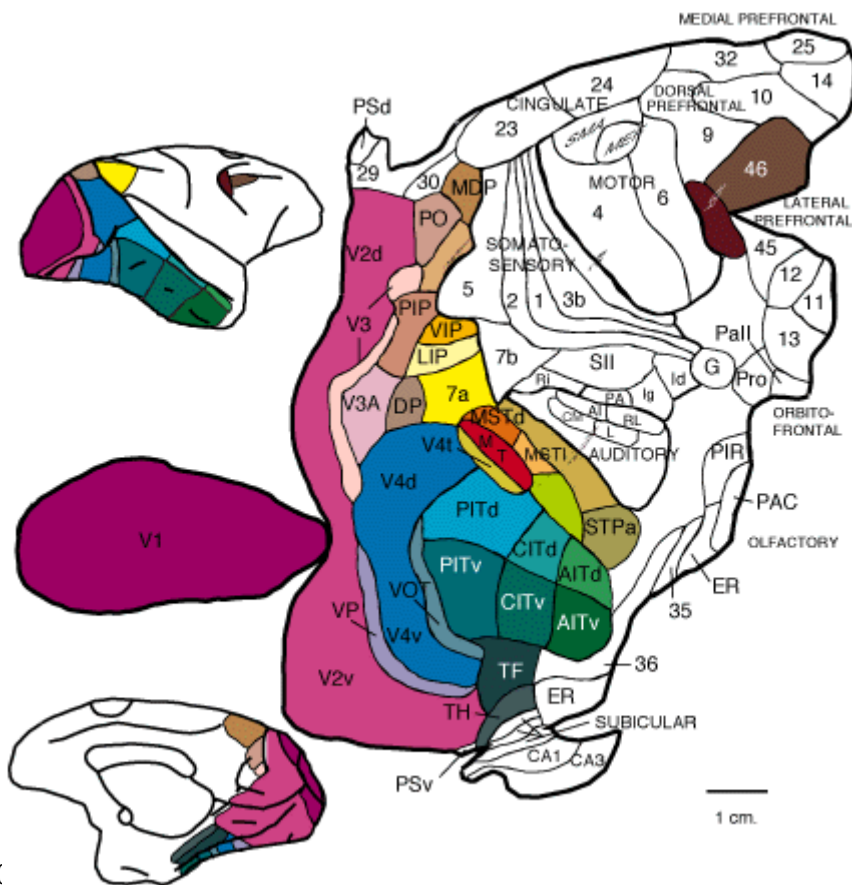
Carnegie Mellon
Spring 2011

Why Don't Computers See Very Well?

Approx. 1/3 of the human brain is devoted to vision!

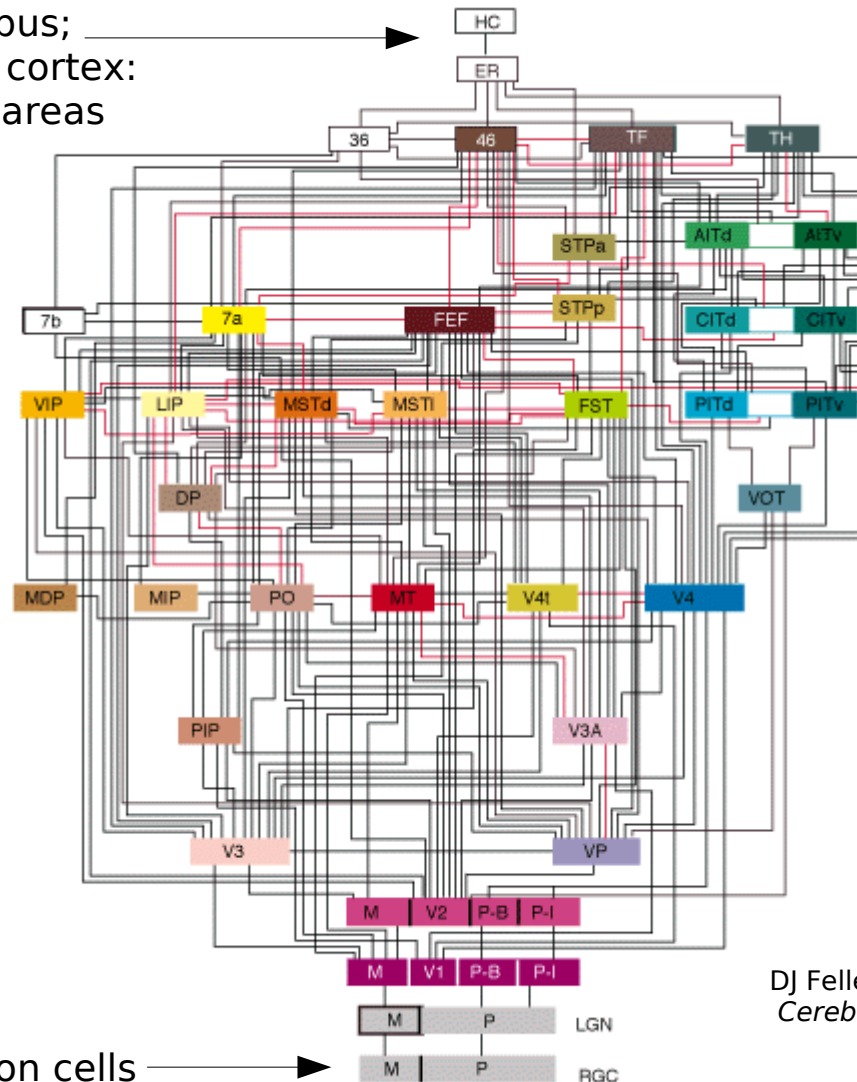
Felleman and Van Essen's Flat Map of the Macaque Brain

DJ Felleman and DC Van Essen (1991), *Cerebral Cortex* **1**:1-47.



The Macaque “Vision Pipeline” as of December 1990

HC = hippocampus;
ER = entorhinal cortex:
high level brain areas



DJ Felleman and DC Van Essen (1991),
Cerebral Cortex **1**:1-47.

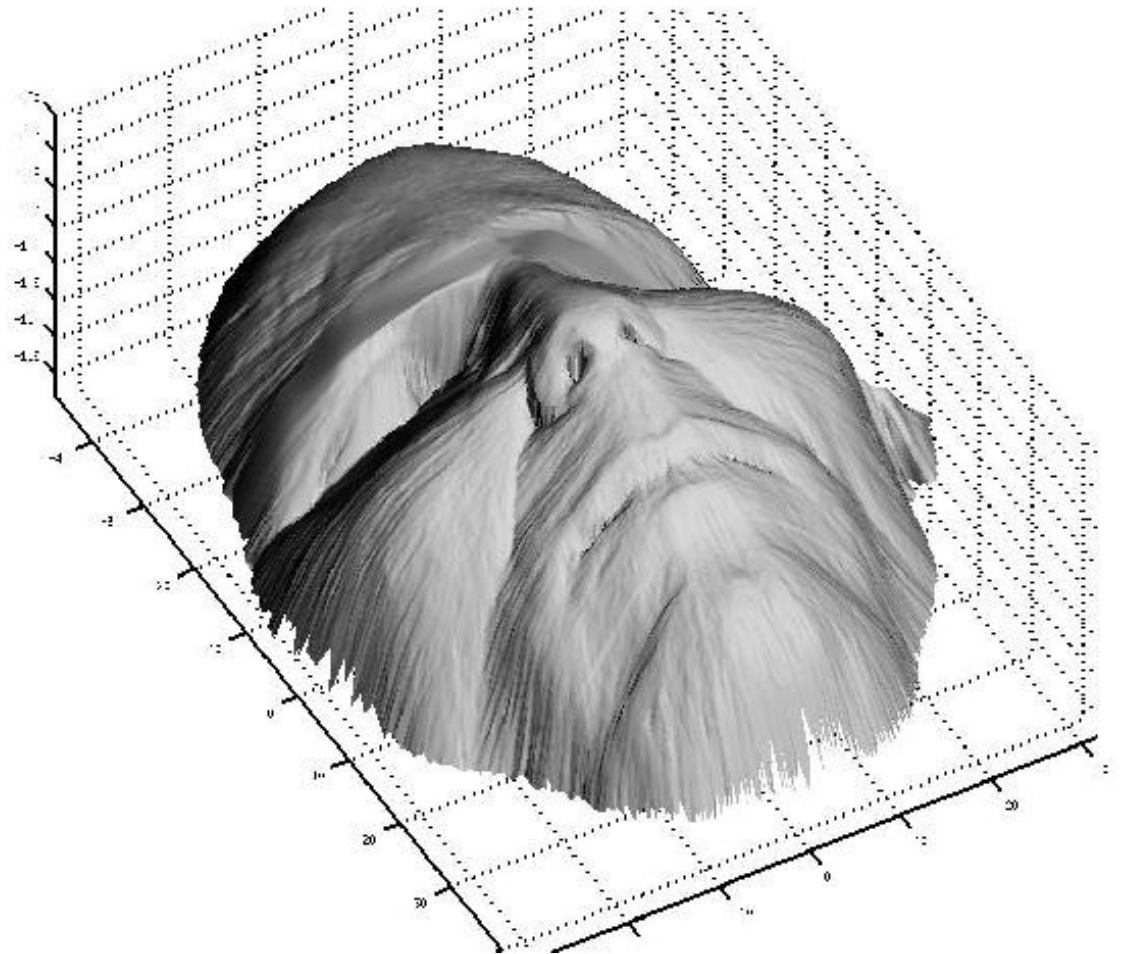
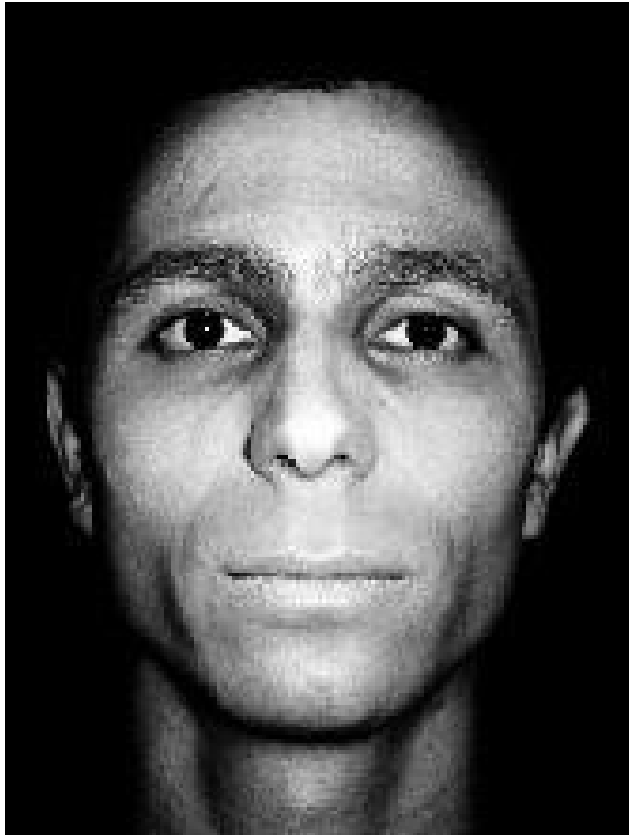
Why Is Vision Hard?

- Segmentation: where are the boundaries of objects?
- Need to recover 3-D shapes from 2-D images:
 - Shape from shading
 - Shape from texture
- Need to fill in occluded elements – what aren't we seeing?
- Importance of domain knowledge:
 - Experience shapes our perceptual abilities
 - Faces are *very* special; there are “face cells” in IT (inferotemporal cortex)
 - Reading is also special; learning to read fluently alters the brain

The Segmentation Problem



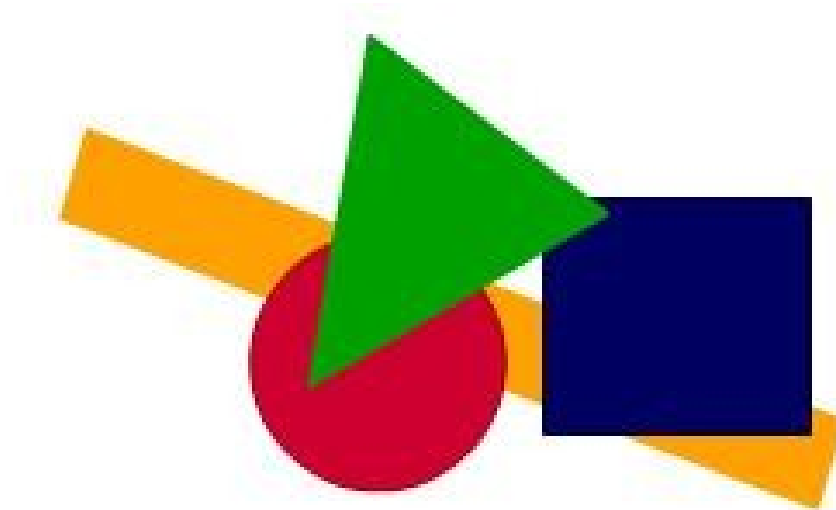
Shape From Shading



Images from: www.cs.ucla.edu/~eprados/

Occlusion

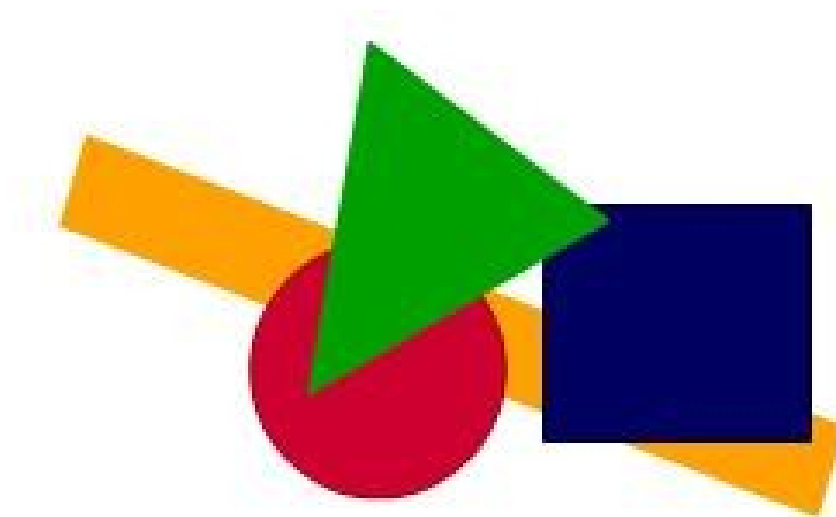
- How many *rectangles* can you find?



- What shapes are present in the image?

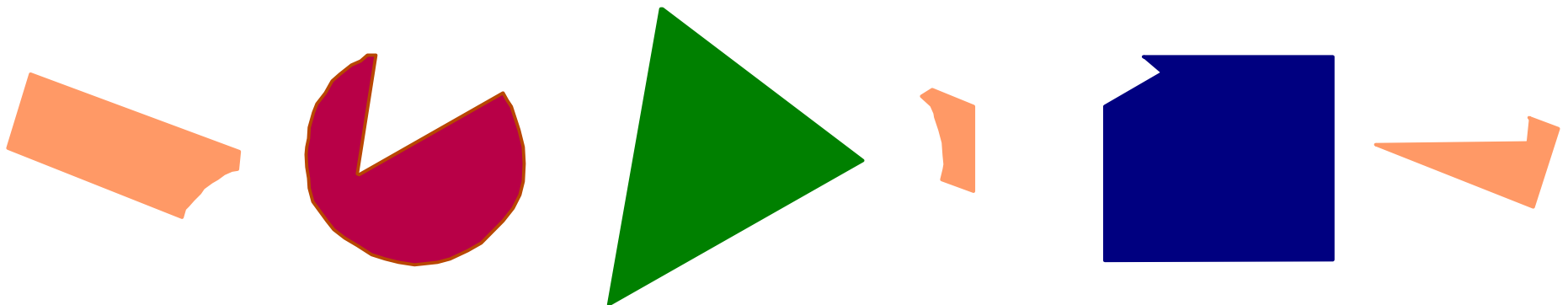
Occlusion

- How many *rectangles* can you find?



None! (Or two.)

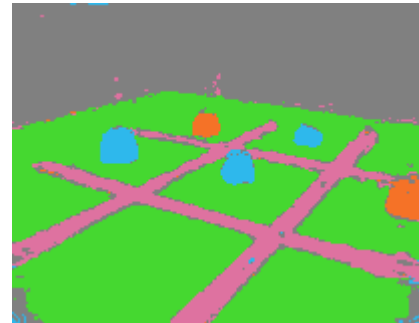
- What shapes are present in the image?



Vision is Hard!

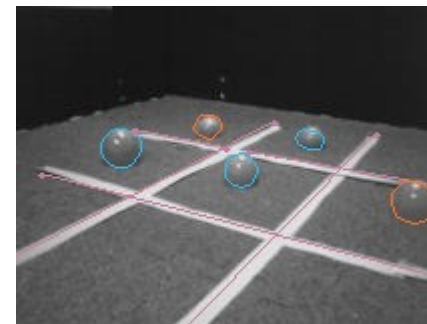
How Can a Poor Robot Cope?

- Use color to segment images.
- Discard shading and texture cues.



From colors to objects:
green = floor
pink = board
blue, orange = game pieces

- Planar world assumption
(can be relaxed later).
- Domain knowledge for occlusion
(blue/orange occludes pink.)



What is “Color” ?

- Humans have 3 types of color receptors (cones).
- Dogs have 2: they're red/green colorblind.
- Cats have 3, but sparse: weak trichromants.
- Birds have 4 or 5 types.
- Birds and honeybees can see ultraviolet; honeybees can't see red.
- Rats lack color vision.

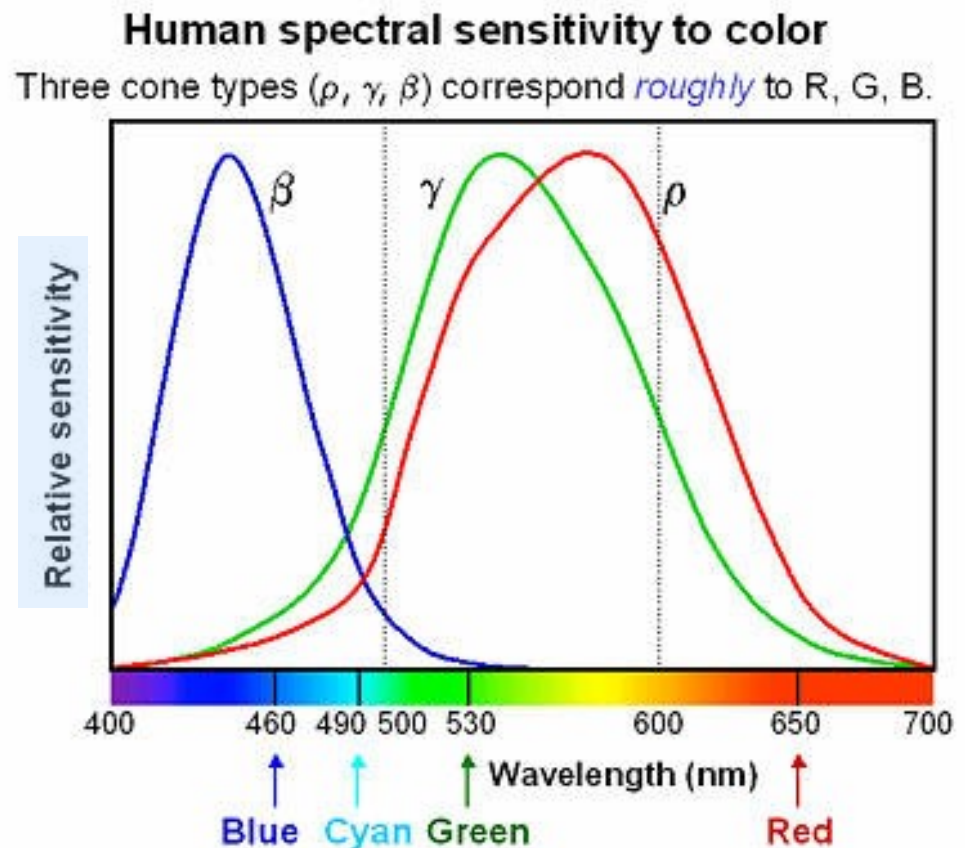
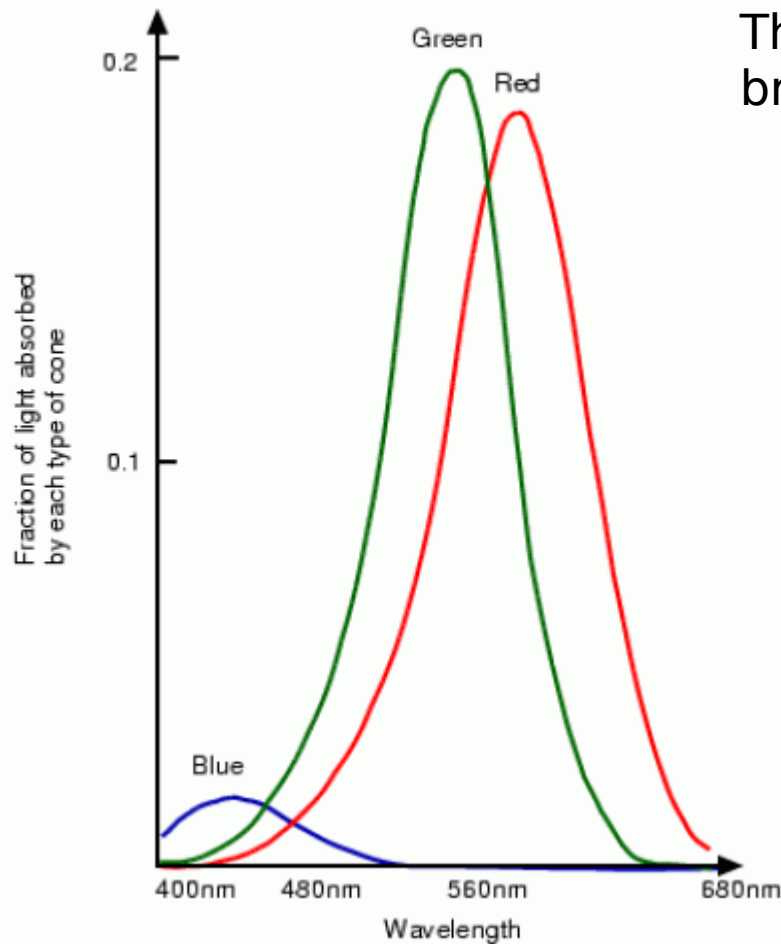
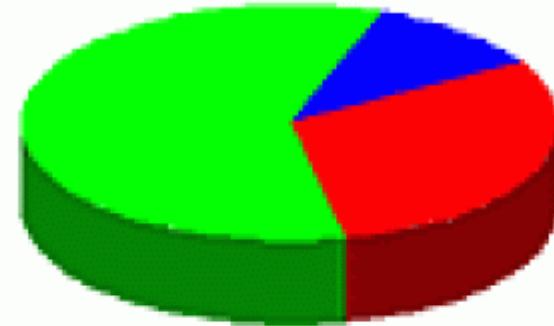


Image from:
http://www.normankoren.com/Human_spectral_sensitivity_small.jpg

The Human Retina is Most Responsive to Green Light



That's why green laser pointers look brighter than red ones of the same power.



"Greyscale"

$$Y = 0.30 * R + 0.59 * G + 0.11 * B$$

Images from http://www.cse.lehigh.edu/%7Espletzer/cse398_Spring05/lec002_CMVision.pdf

Color and Computers

- Video cameras don't see color the same way the human eye does:
 - Different spectral sensitivity curves.
- Colors that look different to you may look the same to a computer that sees through a camera, and vice versa.
- Computer monitors try to synthesize colors by blending just three frequencies: red(ρ), green(γ), and blue(β).
- No computer monitor can produce the full range of color sensations of which humans are capable.

RGB Color Space

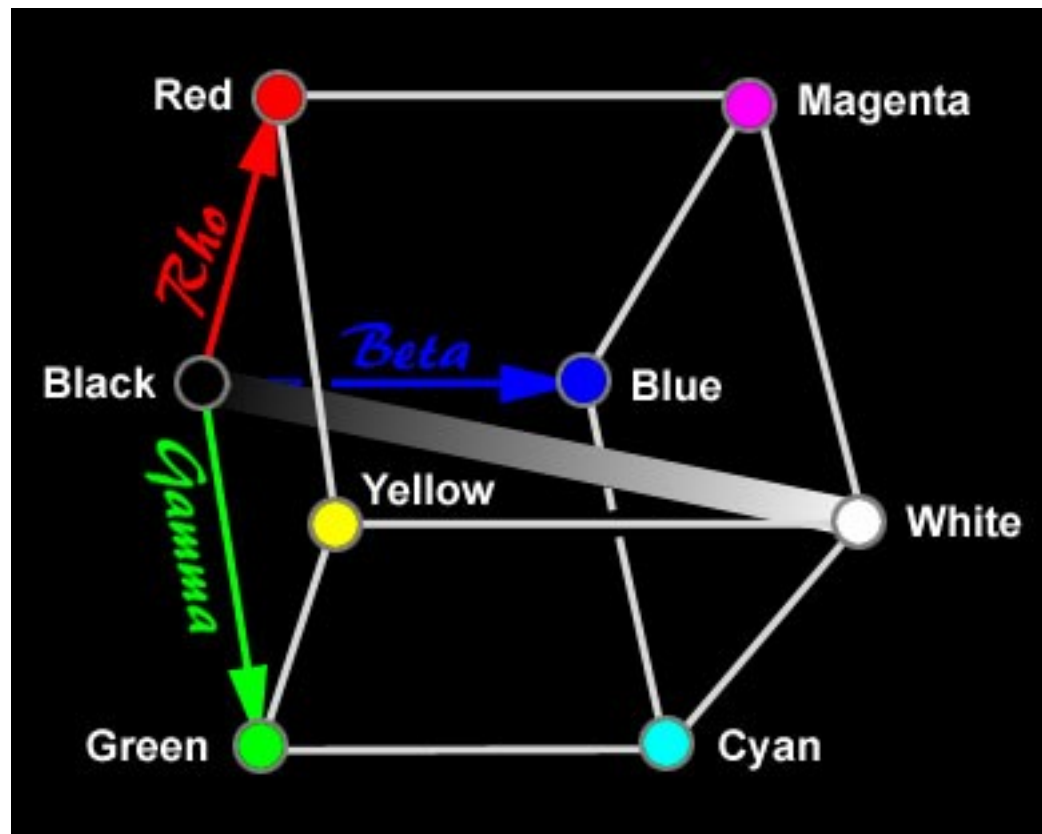


Image from <http://www.photo.net/learn/optics/edscott/vis00020.htm>

Edge of Fully Saturated Hues

Move from one corner to the next by increasing or decreasing one of the three RGB components.

Example: moving...

From green to yellow:

$[0, 255, 0] \rightarrow [255, 255, 0]$

From yellow to red:

$[255, 255, 0] \rightarrow [255, 0, 0]$

From red to magenta:

$[255, 0, 0] \rightarrow [255, 0, 255]$

Saturation in RGB space =
 $\max(r, g, b) - \min(r, g, b)$

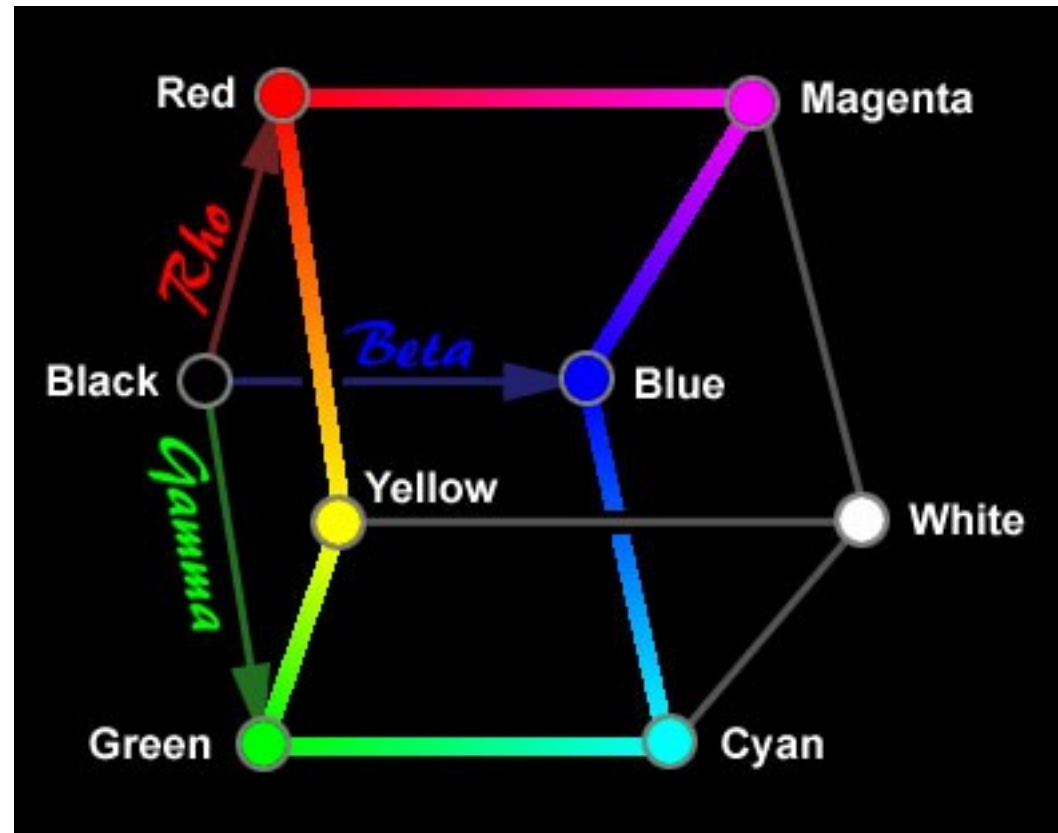
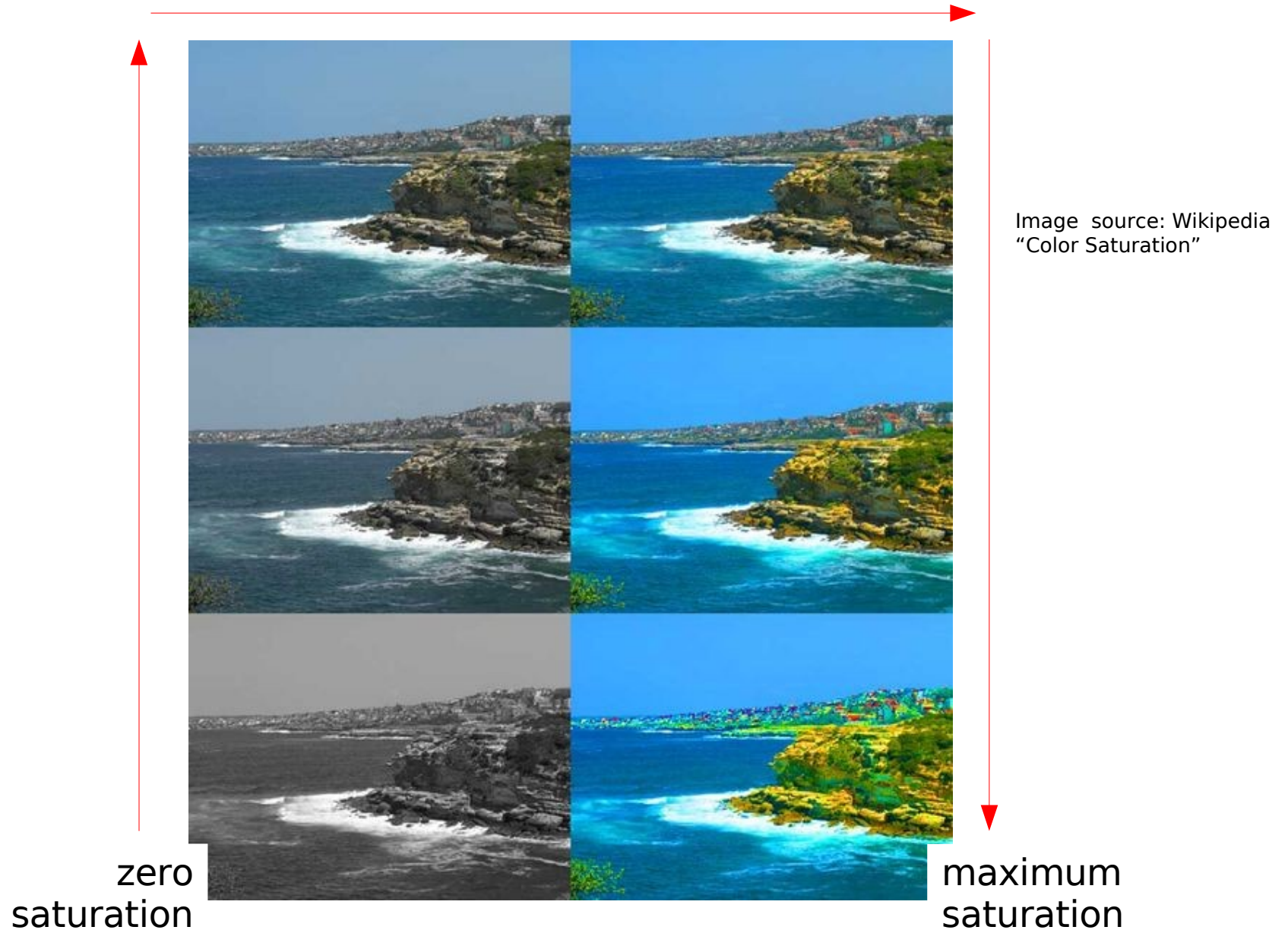


Image from <http://www.photo.net/learn/optics/edscott/vis00020.htm>

Saturation in Images



YUV / YCbCr Color Space

- Y = intensity
- U/Cb = “blueness”
(green vs. blue)
- V/Cr = “redness”
(green vs. red)

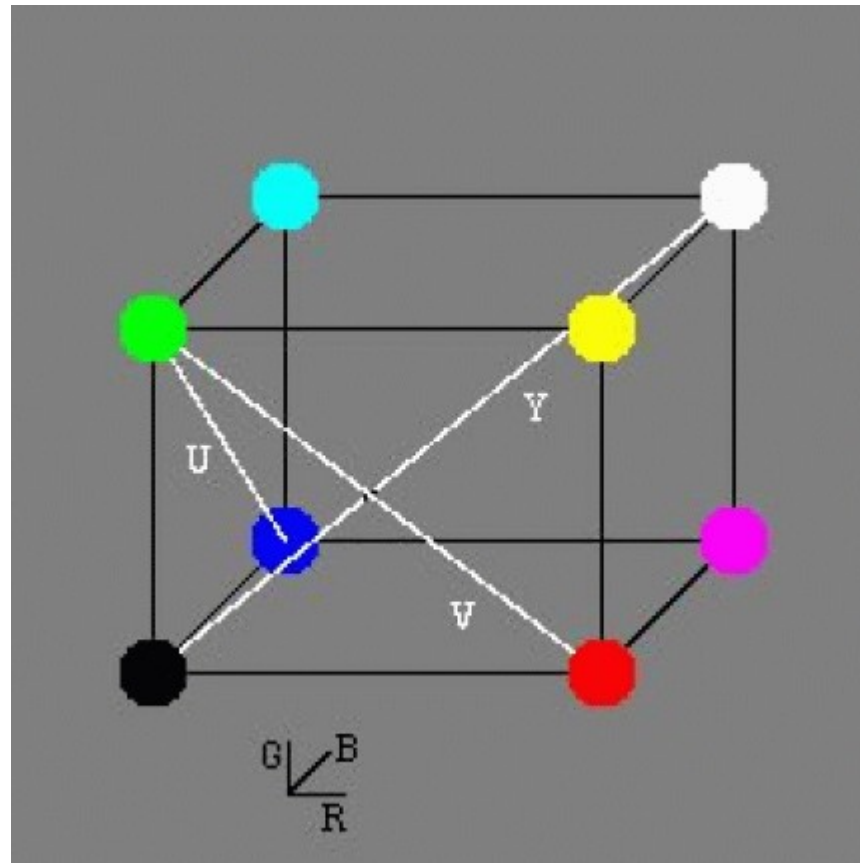
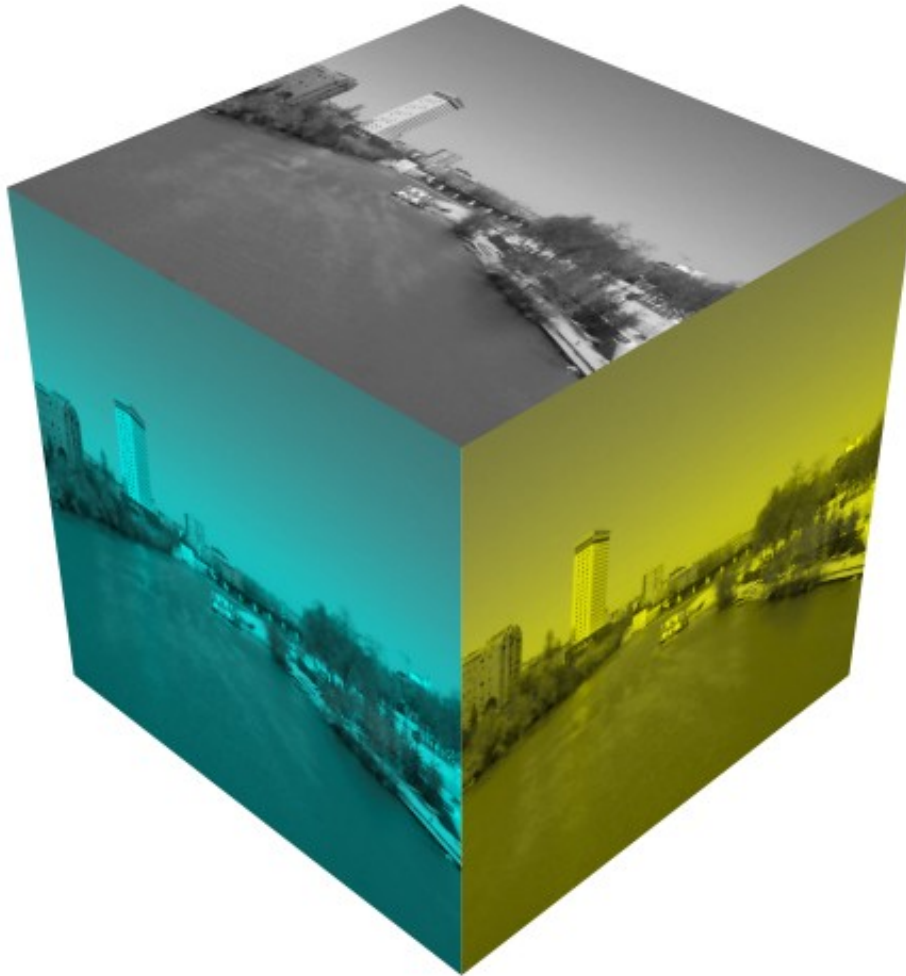


Image from http://www.andrew.cmu.edu/course/15-491/lectures/Vision_I.pdf

YUV Color Cube



Images from http://commons.wikimedia.org/wiki/Image:Cubo_YUV_con_las_capas_de_color.png

Converting RGB to YUV (assuming 8 bits per channel)

$$\begin{bmatrix} Y \\ U \\ V \end{bmatrix} = \frac{1}{256} \cdot \begin{bmatrix} 65.738 & 129.057 & 25.064 \\ -37.945 & -74.494 & 112.439 \\ 112.439 & -94.154 & -18.285 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} + \begin{bmatrix} 16 \\ 128 \\ 128 \end{bmatrix}$$

HSV Color Space

- H = hue
- S = saturation
- V = value (intensity)

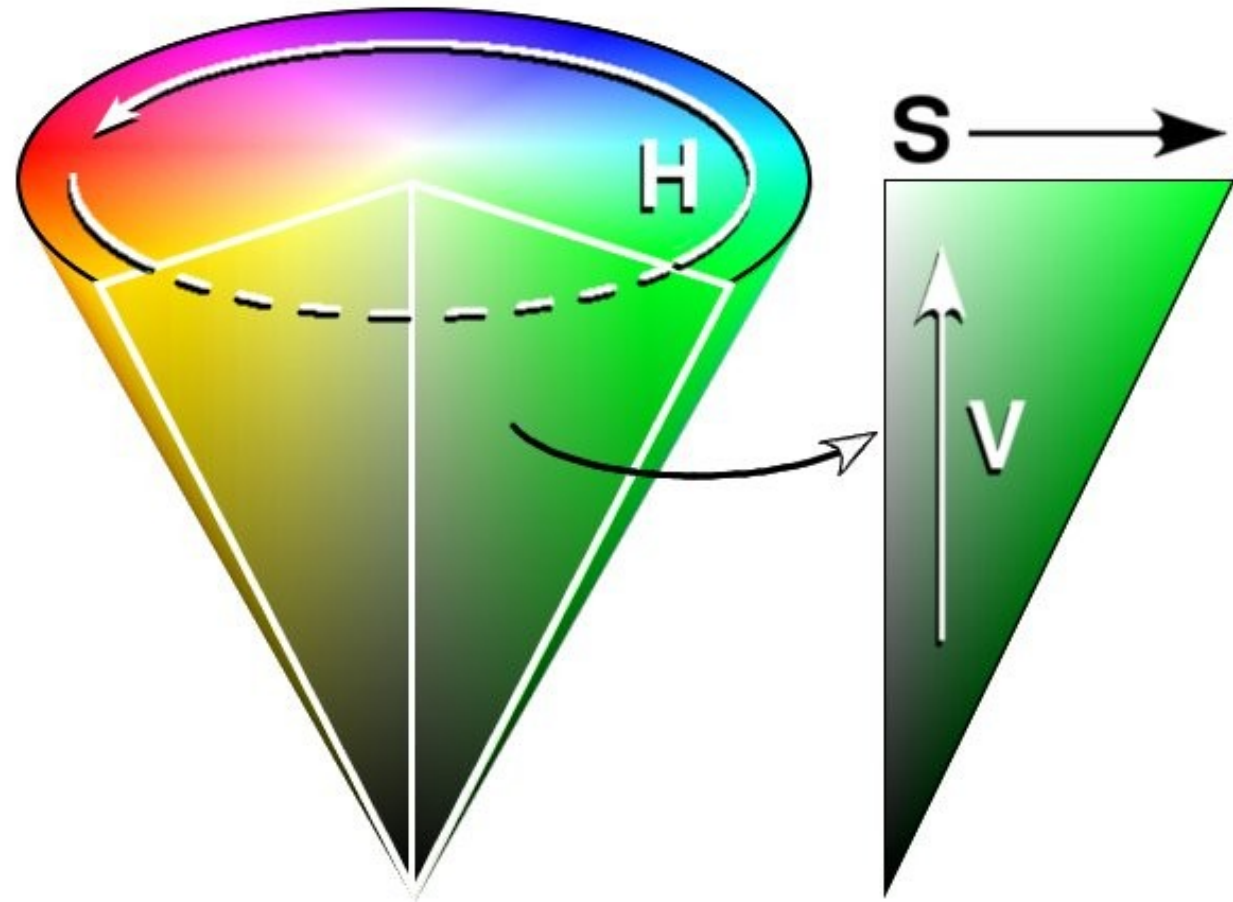


Image from http://www.wordiq.com/definition/Image:HSV_cone.jpg

Many Cameras Use YUV

What the robot sees



What is displayed
for humans



Segmented image



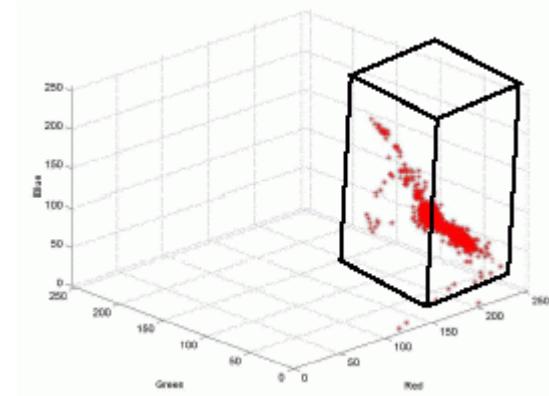
Color Classification 1

- Define a set of color classes: “pink”, “orange”, etc.
- Each class is assigned some region of color space.

- Simplest case: use rectangles.

```
isOrange[i] =  
    imR[i] >= orangeMinR && imR[i] <= orangeMaxR &&  
    imG[i] >= orangeMinG && imG[i] <= orangeMaxG &&  
    imB[i] >= orangeMinB && imB[i] <= orangeMaxB;
```

- Drawbacks: (1) the “real” regions aren't rectangular, so errors result; (2) lots of colors = slow processing.



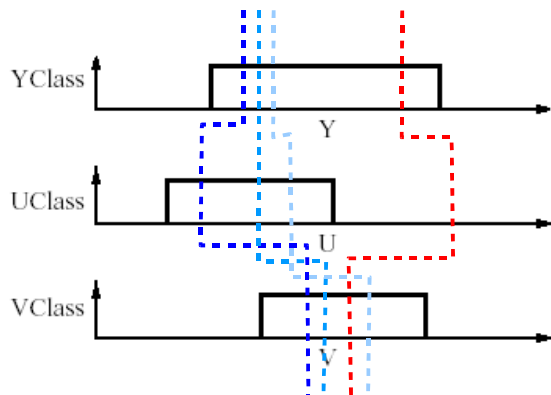
Color Classification 2

- We can have arbitrary-shaped color regions by creating a lookup table.
- For each (R,G,B) value, store the color class (integer).
- Problem: 24 bit color = 16 million entries = 16 MB.
- Could use fewer bits, but that would reduce accuracy.

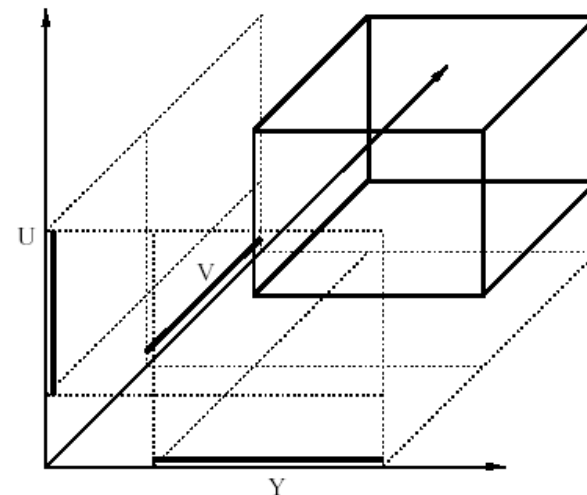
Color Classification 3

- J. Bruce, T. Balch, and M. Veloso, IROS 2000:
- Table lookup with bit-wise AND function can handle 32 color classes at once.

```
int Ytable[256], Utable[256], Vtable[256];  
ColorClasses[i] =  
    Ytable[imY[i]] & Utable[imU[i]] & Vtable[imV[i]];
```



Binary Signal Decomposition of Threshold



Visualization as Threshold in Full Color Space

Bruce et al. (continued)

- We assigned a bit to each color:

1000 = “pink”

0100 = “orange”

0010 = “blue”

0001 = “green”

- Suppose the “pink” and “orange” classes both include some colors with a Y value of 214:

Ytable[214] = 0x1100

- Suppose all four classes include a U value of 56:

Utable[56] = 0x1111

- If “orange” and “green” both include V values of 118:

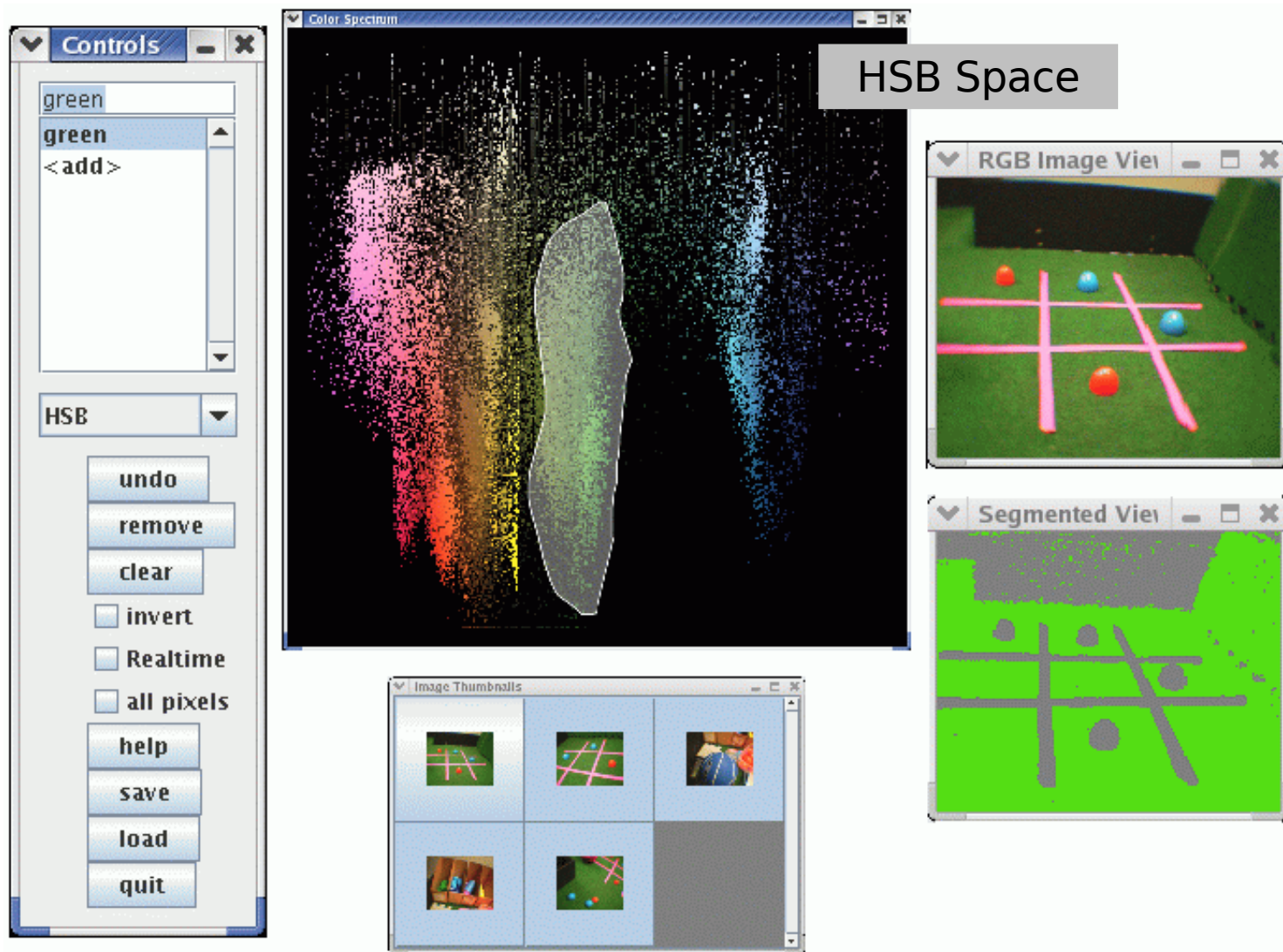
Vtable[118] = 0x0101

- Color classes of (214,56,118) are: 0x0100 = orange

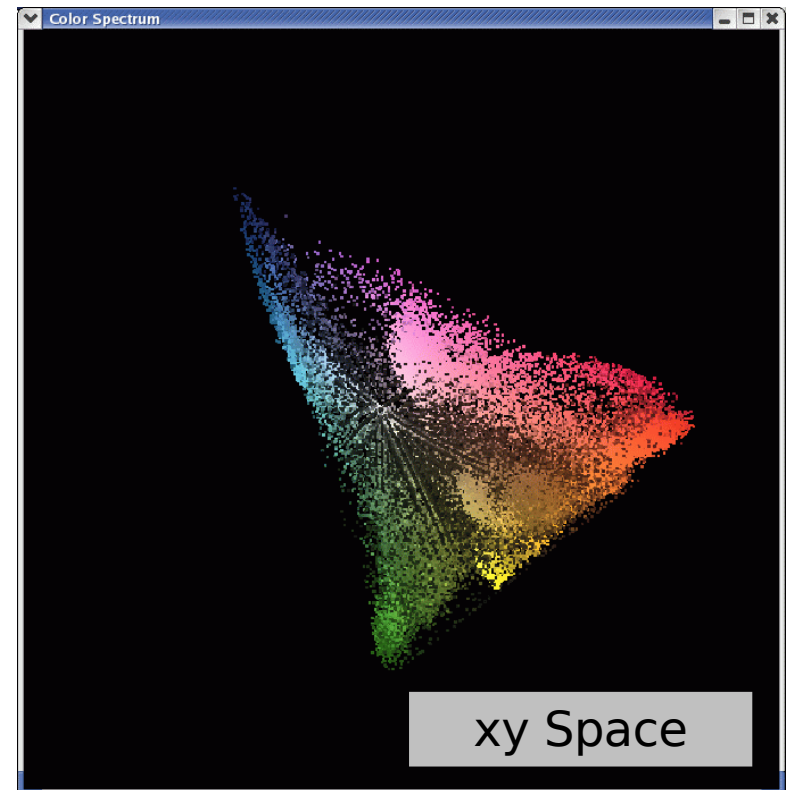
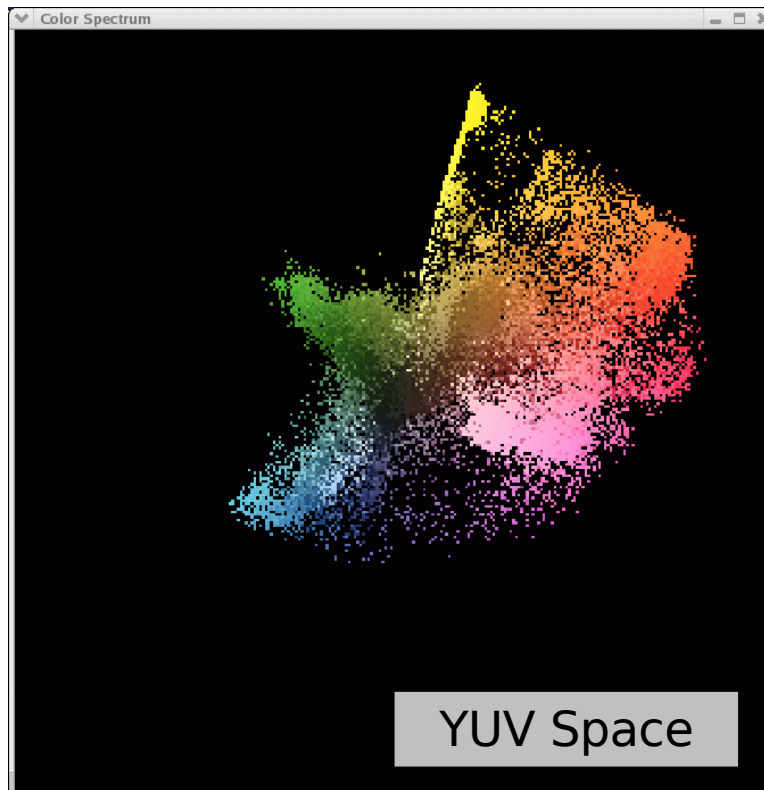
Color Classification 4: CMVision

- CMVision is a vision package developed by Jim Bruce, Tucker Balch, and Manuela Veloso at Carnegie Mellon. Used for many robotics projects.
- Current implementation operates in YUV space. Uses a reduced-resolution lookup table so it's not limited to rectangular decision boundaries.
 - 4 bits for Y, 6 bits each for U and V: 65,536 entries.
- The format of a CMVision threshold map (.tm) file is:
TMAP
YUV8
16 64 64
<65,536 1-byte table entries>

The EasyTrain Tool Creates Threshold Files for CMVision

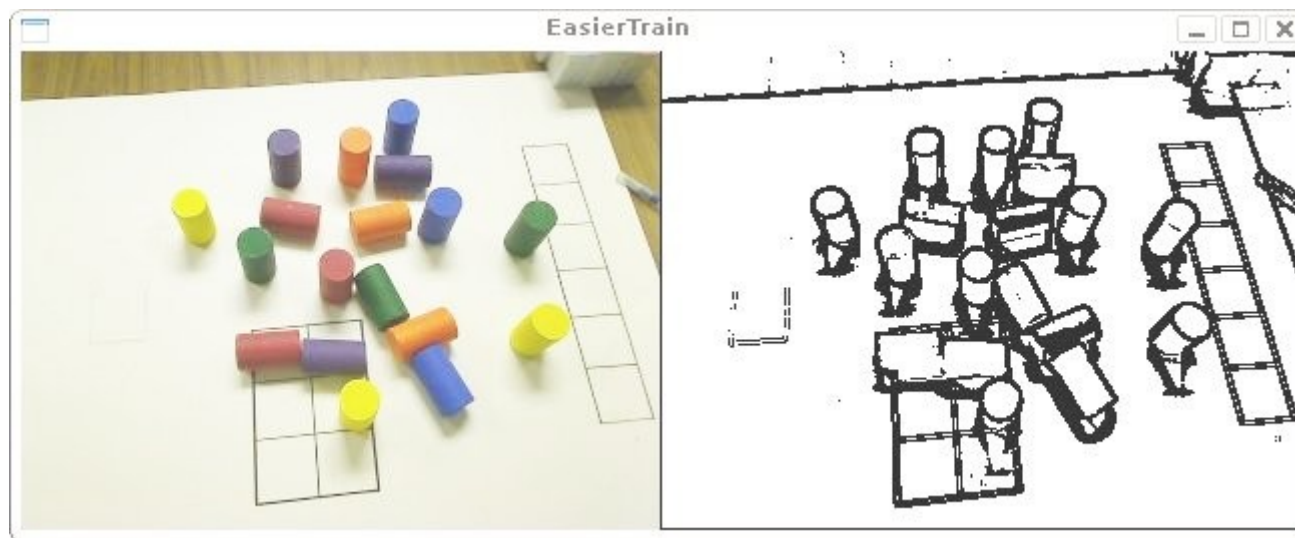


Other Color Spaces Supported

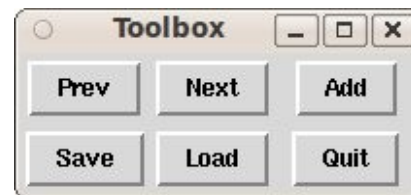
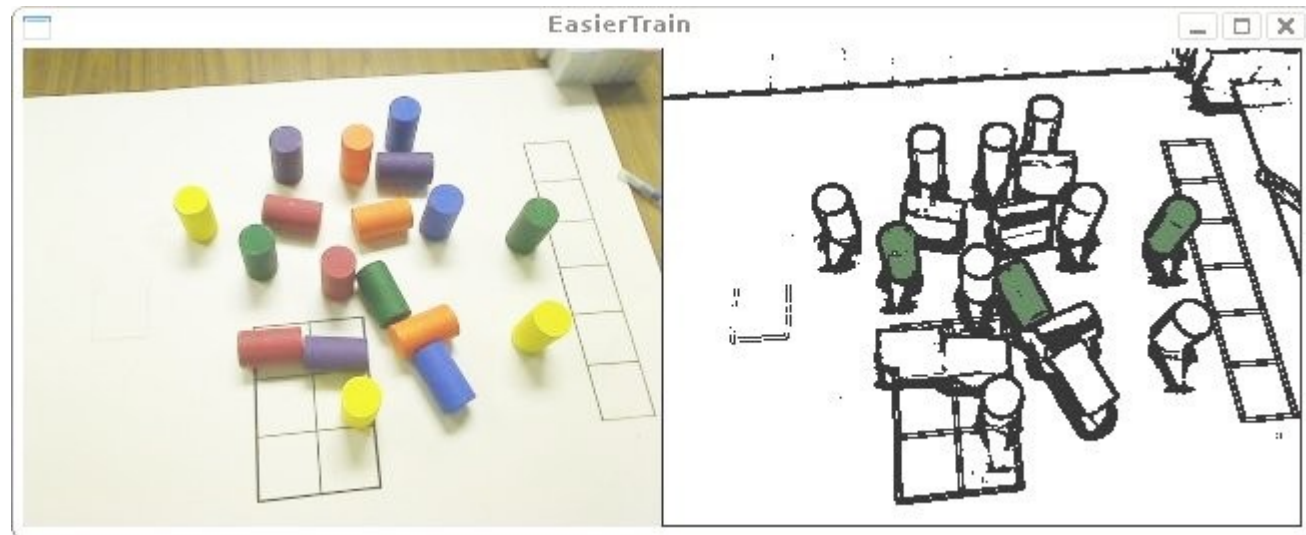
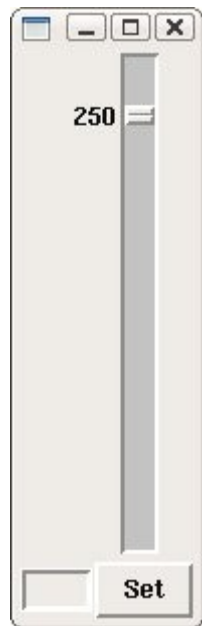


EasierTrain

- Created by Michael Gram and Nathan Hentoff at RPI.
- <http://code.google.com/p/tekkotsu-easiertrain>
- Automatically segments the image and allows the user to assign color names and adjust segmentation thresholds.

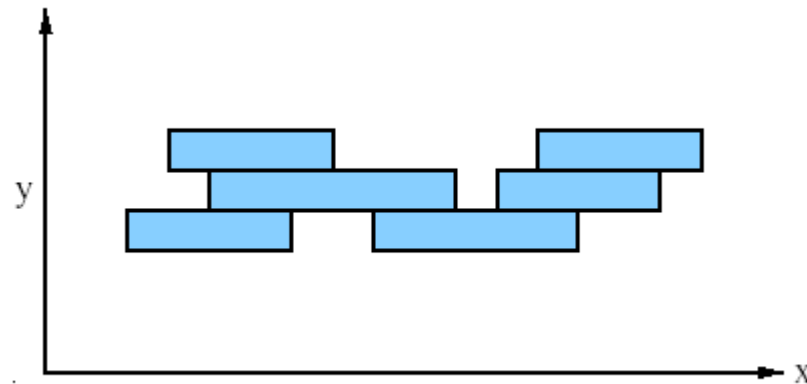


EasierTrain



Run Length Encoding

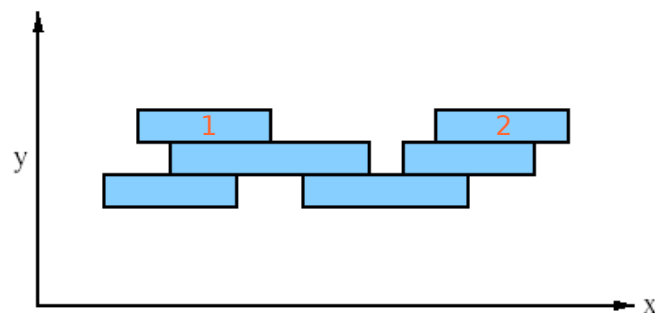
- Next step after color segmentation.
- Replace identical adjacent pixels by run descriptions:
 - Lossless image compression.
- An image is now a list of *rows*.
A *row* is a list of *runs*, of form:
<starting column, length, color class>



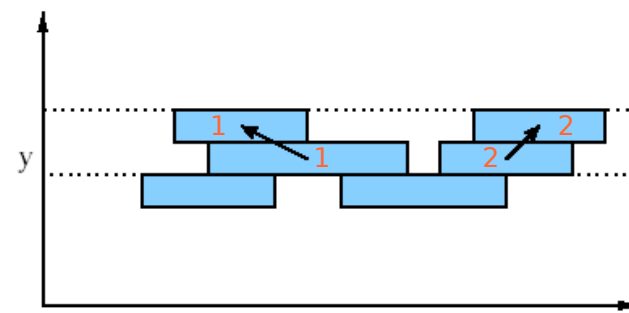
- Run length encoding also does noise removal, by skipping over short gaps between runs.

Connected Components Labeling

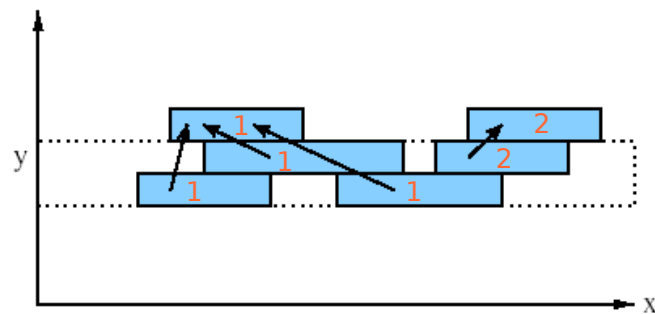
- Assemble adjacent runs of the same color into regions.
- This gives crude object recognition, assuming *that identically-colored objects don't touch*.



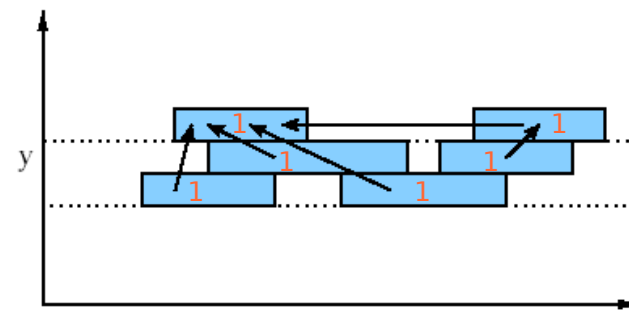
1: Runs start as a fully disjoint forest



2: Scanning adjacent lines, neighbors are merged



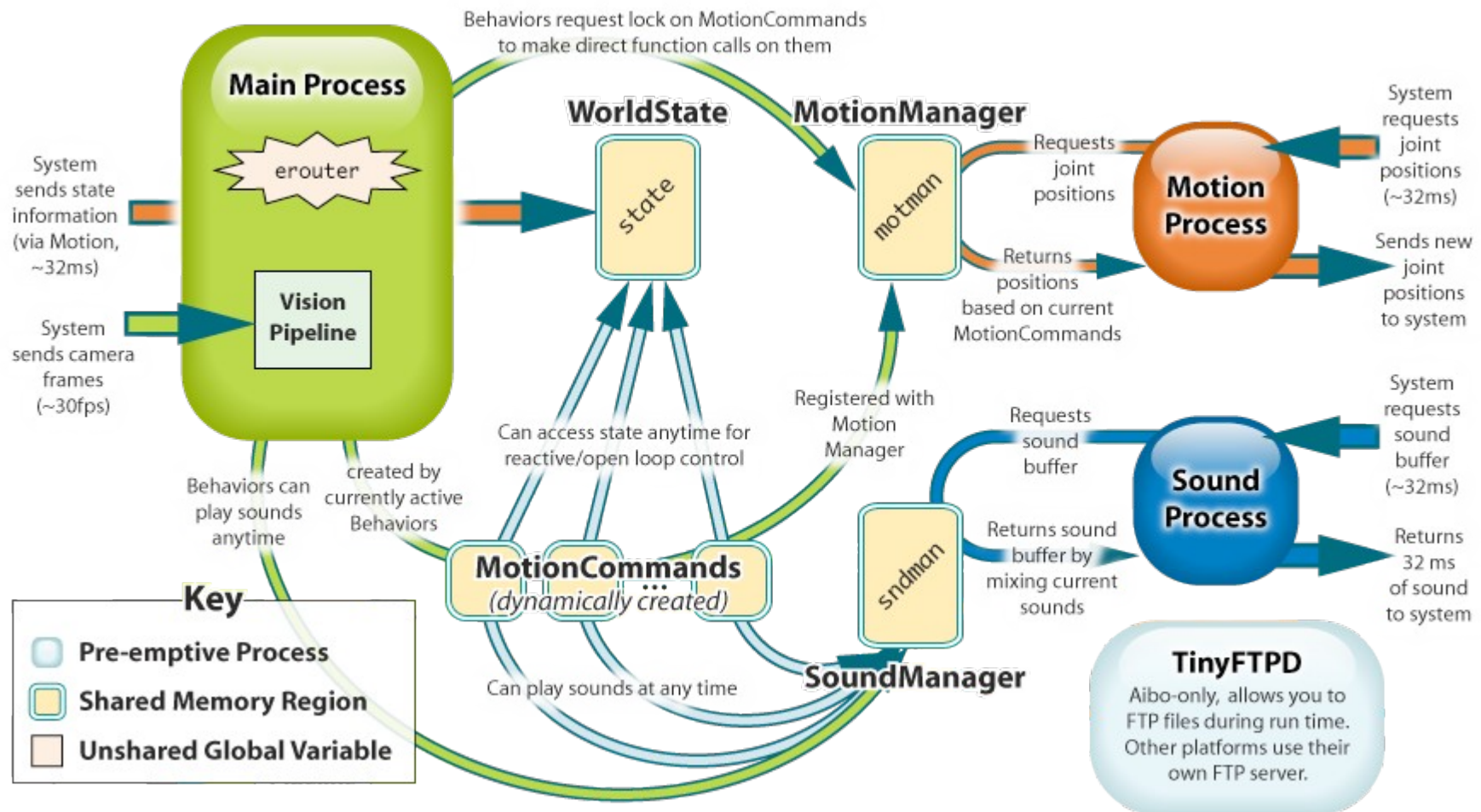
3: New parent assignments are to the furthest parent



4: If overlap is detected, latter parent is updated

Image from
Bruce et al.,
IROS-2000

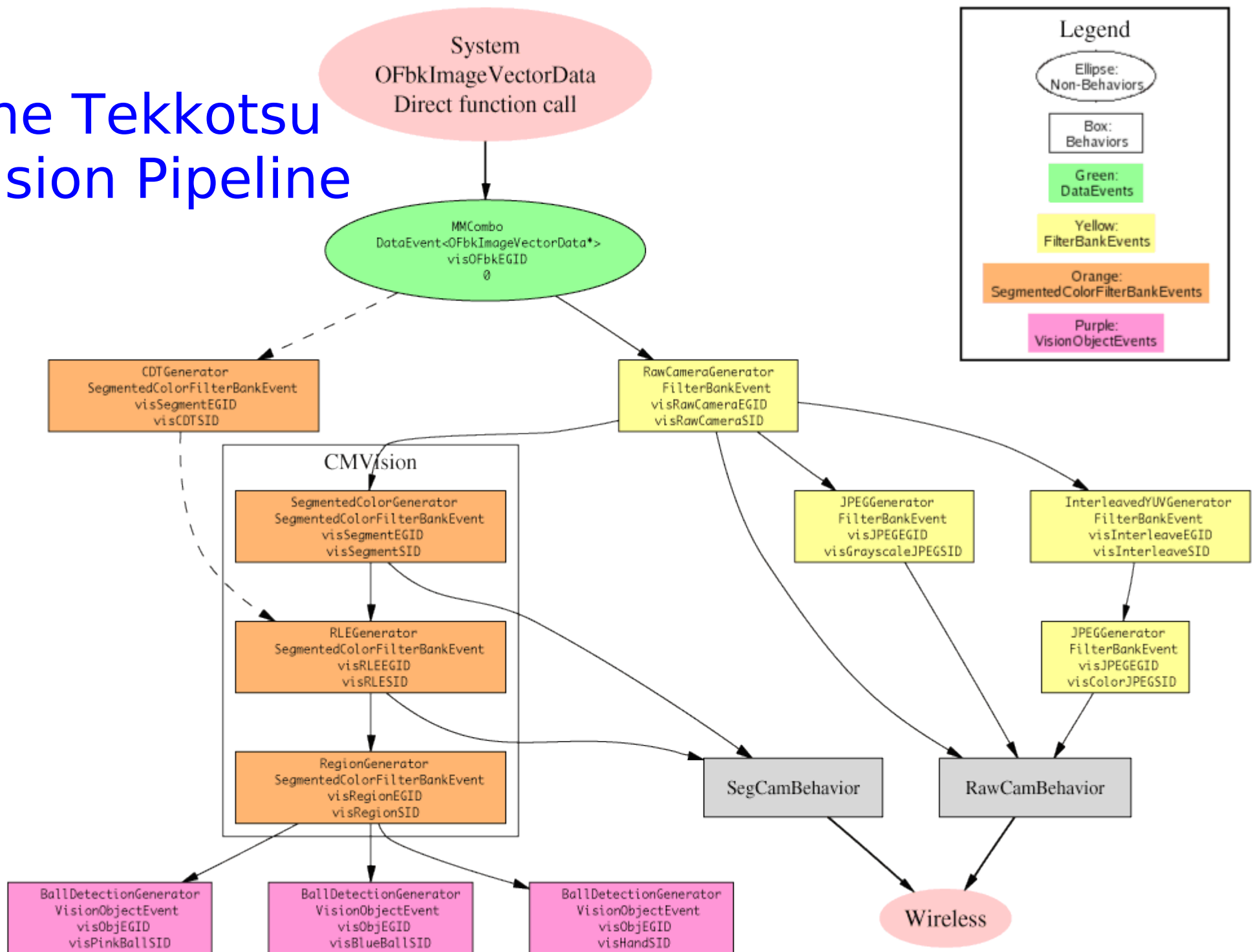
Tekkotsu Vision is Done in the Main Process



Tekkotsu Vision Pipeline

- CDTGenerator: color detection table (AIBO); unused
 - SegmentedColorGenerator
 - Color classified images
 - RLEGenerator
 - Run Length Encoding
 - RegionGenerator
 - Connected components
 - BallDetectionGenerator
 - Posts VisionObjectEvents for largest region if shape is roughly spherical
 - DualCoding Representations / MapBuilder
- 
- CMVision

The Tekkotsu Vision Pipeline



Tekkotsu Vision Pipeline

- Image pyramid: double, full, half, quarter, eighth, and sixteenth resolution streams are available.
- Six channels available: Y, U, V, Y_dx, Y_dy, Y_dxdy. (The latter three are for edge detection.)
- Lazy evaluation: generators only run if some behavior has subscribed to their events.
- RawCameraGenerator and JPEGGenerator feed RawCamBehavior (for ControllerGUI RawCam viewer)
- SegCamBehavior uses RLE encoded images

Summary of Vision in Tekkotsu

- Simple blob detection using VisionObjectEvent (reports largest roughly spherical blob of a specified color)
- Dualcoding representations:
 - Sketches (pixel representation)
 - Shapes (symbolic representation)
 - Lookout, MapBuilder
- Object recognition using SIFT
- AprilTags (implementation of Augmented Reality Tags)

