Cameras, Images, and Low-Level Robot Vision

RSS Lecture 4

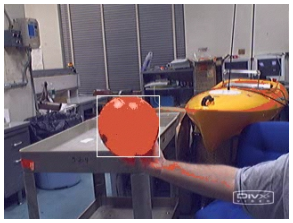
Tuesday, 18 Feb 2014

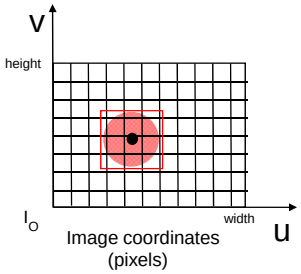
Prof. Teller

Siegwart and Nourbakhsh § 4.1.8

Digital image contents

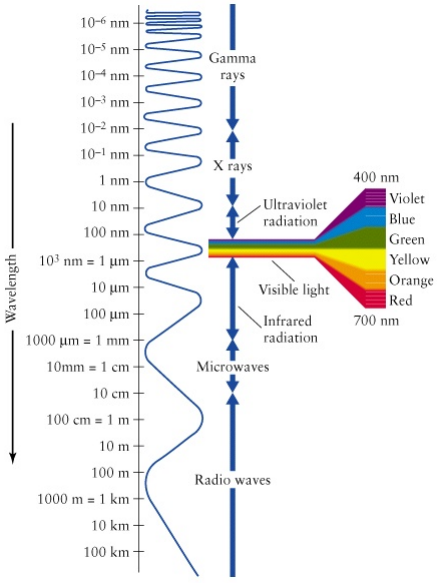
- Why are pixels represented as “RGB”?
  - Is world made of red, green, and blue “stuff”?



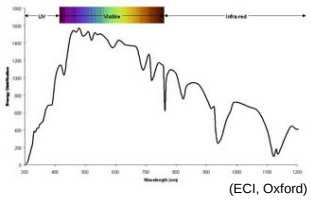


- … Answer requires two brief digressions about human vision & cameras as sensors

Visible light spectrum

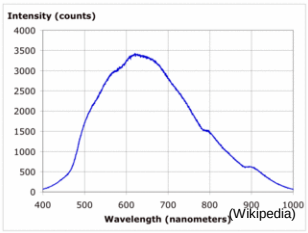


(Freedman & Kaufmann, Universe)



(ECI, Oxford)

Solar spectrum



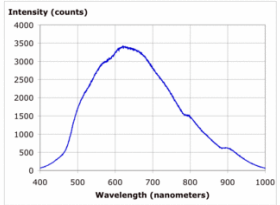
(Wikipedia)

Incandescent spectrum

Image as measurement

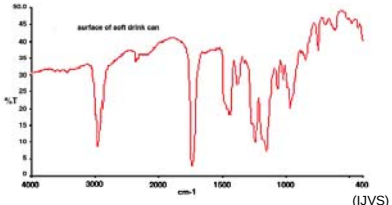
- What does eye/camera actually *observe*?
  … the *product* of illumination spectrum with absorption or reflection spectrum!

= (at each image point)



Illumination spectrum

x

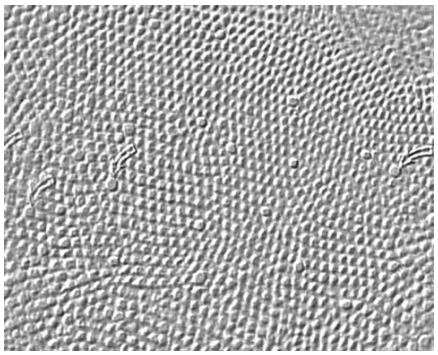


Reflection spectrum

(IJVS)

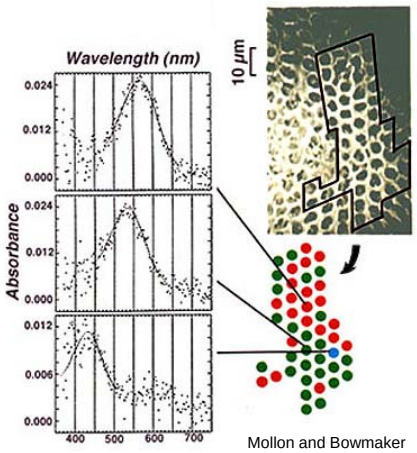
Foveal cone distribution

- Densely packed in fovea, less so in periphery



Visual discrimination of *1 minute of arc* [corresponds roughly to] the center-to-center spacing (3 µm) of the cones of the central mosaic in the foveola (retina.umh.es).

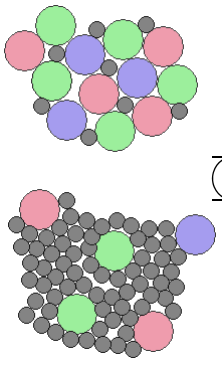
What does “1 minute of arc” mean?



Mollon and Bowmaker

Cone sensitivities

- Three cone types (S, M, and L) are roughly blue, green, and red sensors, respectively. Their peak sensitivities occur at ~430nm, 560nm, and 610nm for an "average" human.

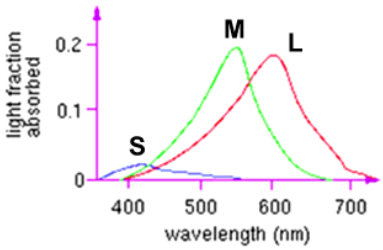


Rods & cones, ~1.35 mm from center of fovea

4 µm

Rods & cones, ~8 mm from center of fovea

(IJVS)



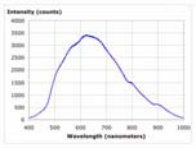
Cone sensitivities as a function of wavelength

Color perception

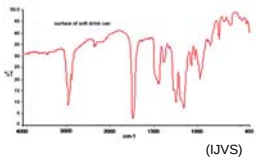
- The cones form a spectral “basis” for visible light; incident spectral distribution differentially excites S,M,L cones, leading to color vision

=

(at each cone site)

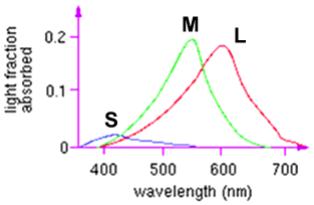


X



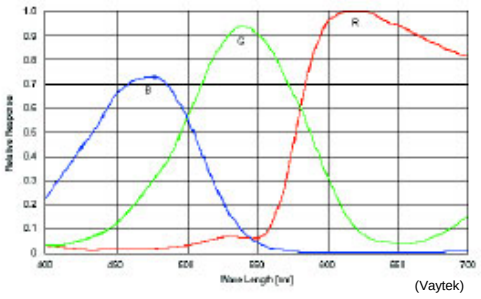
(IJVS)

X



Origin of RGB CCD sensors

- So, in a concrete sense, CCD chips are designed as RGB sensors in order to emulate the human visual system

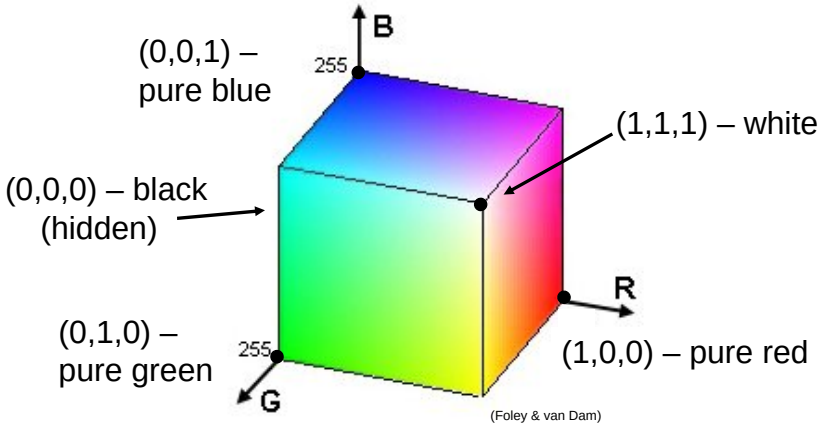


CCD with Bayer Filter, Relative Spectral Response Curve

- … End of digressions

RGB Color Model

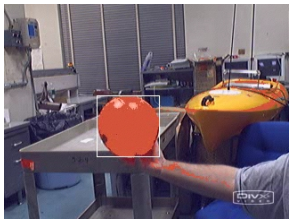
- Think of R, G, B as a kind of “color orthobasis”

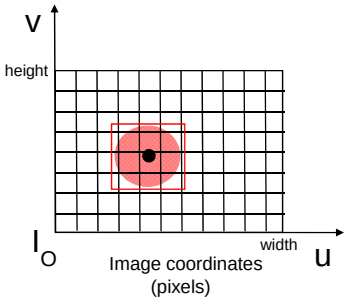


(Foley & van Dam)

Object detection

- Suppose we want to detect an object (e.g., a red ball) in camera’s field of view





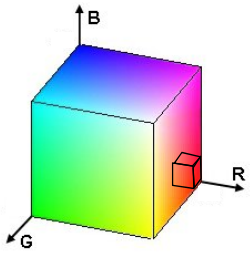
- We simply need to identify all pixels of some specified color in the image … right?

Naïve object detector

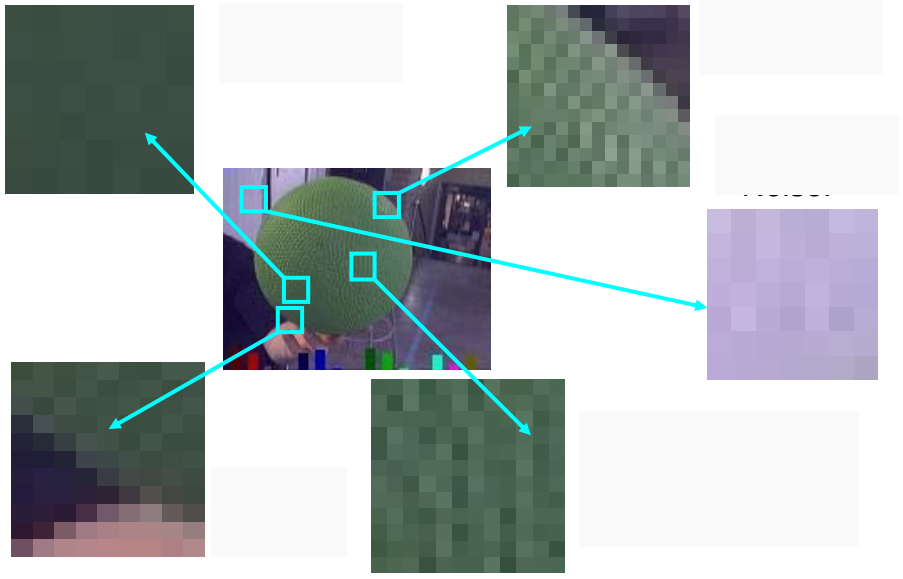
```
set objectPixels = Ø; // empty set

// look for red ball in image
for i = 0 to width-1
   for j = 0 to height-1
      if ( isRed( pixel[i, j] ) ) // classifier
         objectPixels = objectPixels U {(i, j)};

if ( isBall ( objectPixels ) ) // detector
   // do something in response to ball
```

Pixel classification

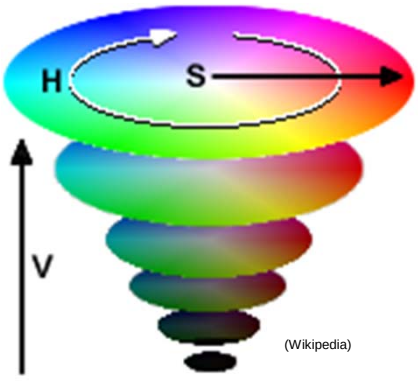
```
Boolean isRed ( pixel p ) {
   if (      p.red >= 0.8     // where do 0.8,
        && p.green < 0.2   // 0.2 come from?
        && p.blue < 0.2 )
      return true;
   else
      return false;
}

// Will this do what we want?
```



Confounding effects: Real-world images



Alternative: HSV Color Model

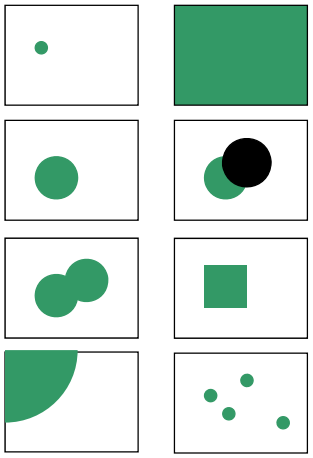


(Wikipedia)

- More robust under illumination changes (why?)
- Still must confront noise, specularity etc.

Naïve object detector

```
Boolean isBall ( set s ) {
  if ( |s| > 0.1 * W * H )  // area threshold
       return true;
  else
       return false;
}

// how might this fail?
```



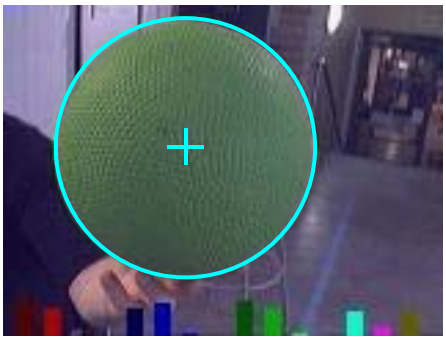
Etc…

(Slightly) improved detector

```
Boolean isBall ( set s ) {
  if ( |s| > 0.1 * W * H  // area threshold
       && s is “ball-shaped” ) {
     return true;
  else
     return false;
}

// how might this fail?
```

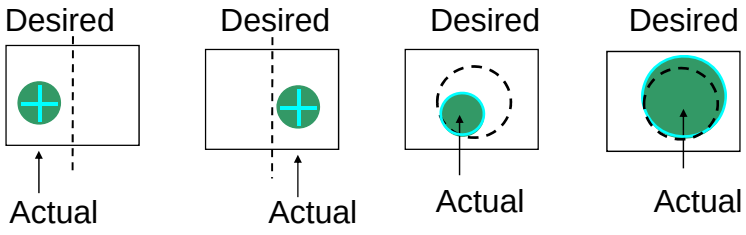
Doing something useful

- Report *presence* of ball in image
  - As function return, message dispatch, etc.
- Estimate *attributes* of (presumed) object
  - Color
  - Size
  - … how?
  - Centroid
  - … how?



- How / when might these estimates be poor?

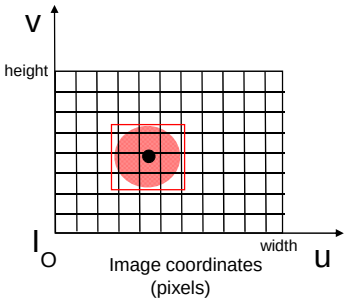
Size, centroid estimation

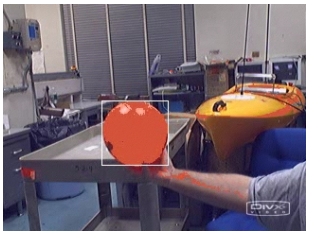


- Can use estimators as inputs to motion controller!
- Suppose we want a 1-meter frontal standoff
  - How do we compute *desired* size in image?
  - Instance of *camera calibration*;
    more examples to come later in term

Application: Visual Servoing (Lab 3)

- Write a “blob detector” in integer (u, v) pixel coordinates
  - Transform pixels from (r,g,b) to chrominance, luminance
  - Given a target hue (e.g., red) and error tolerance, find significant connected components of pixels with that hue
  - Estimate the area and centroid of the largest detected blob
- We will supply several “fiducial objects” (colored balls)
- Issue translation, rotation control so that robot “servos” to the ball, facing it frontally at desired standoff distance





What’s Next in RSS:

- Today’s Lab
  - Continue work on Lab 2
- Wednesday 2/19
  - Lecture (Alec): ROS, the Robot Operating System
  - Briefings for Lab 2 in alcove, teams in random order
  - Lab 3 (Visual Servoing and ROS) out
- Friday 2/21
  - CDE’s returned with technical, communication feedback
  - Forum: Briefing Conferences for Teams 1-4
- Monday 2/24
  - Lecture: Robot sensors and control architectures