

Cameras, Images, and Low-Level Robot Vision

RSS Lecture

M 2 Mar 2009

Prof. Teller

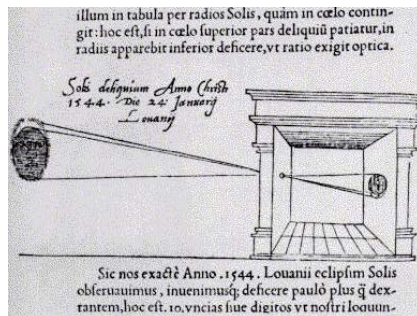
Text: Siegwart and Nourbakhsh S 4.1.8

Today's Lecture

- Brief historical overview
 - From early cameras to digital cameras
- Low-level robot vision
 - Camera as sensor
 - Color representation
 - Object detection
 - Camera calibration
- Putting it all together
 - Visual servo lab (next week)

Camera Obscura

- Mo Ti, Chinese philosopher, 5th Century B.C.
 - Described linear light paths, pinhole image formation
- Leonardo da Vinci (1452-1519)
 - Demonstrated camera obscura (lens added later)
 - Etymology: camera + obscura = “dark room”



Frisius (1544)

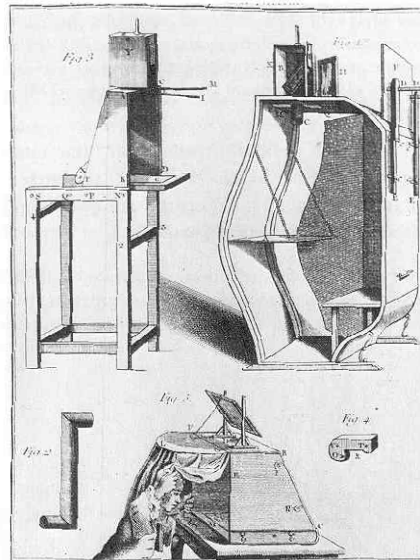
Photograph of camera obscura interior:



Portmerion Village, North Wales

Toward Photography

- People sought a way to “fix” the images at the back of the camera obscura
- Pursued decades of experimentation with light-sensitive salts, acids, etc.
- First photograph produced when?




First Photograph



Harry Ransom Center

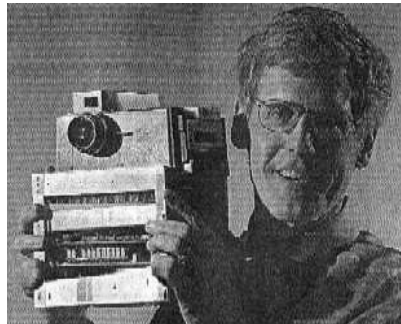
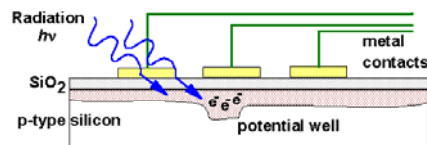


Kodak (reproduction)

- Joseph Nicéphore Niépce (pronounced "Neeps"), "View from the Window at Le Gras," c. 1826 
- Aluminum plate coated with light-sensitive material
- Why are buildings illuminated on both sides?
- Etymology: "photo" + "graph" (also: photogene, heliograph)

Digital Cameras

- Photoelectric effect (Hertz 1887; Einstein 1905)
 - As light *frequency* increases?
 - As light *intensity* increases?
- Charge-coupled devices as storage (late 1960's)
- Light sensing, pixel row readout (early 1970's)
- First electronic CCD still-image camera (1975):
 - Fairchild CCD element
 - Resolution: 100 x 100 b/w
 - Image capture time: 23 sec., mostly writing cassette tape
 - Total weight: 8-1/2 pounds



Kodak, c. 1975

Miniaturization, price point

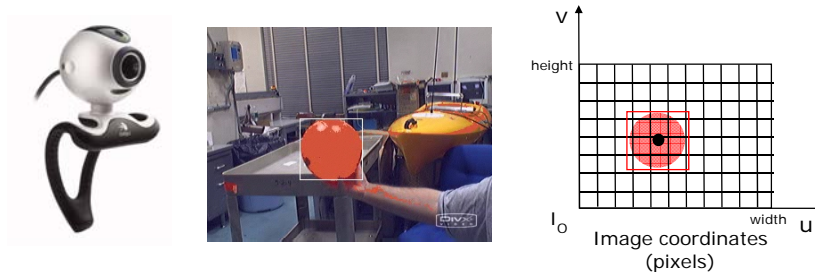
- C. 2005, fifty dollars buys a camera with:
 - 640 x 480 pixel resolution at 30Hz
 - 1280 x 960 still image resolution
 - 24-bit RGB pixels (8 bits per channel)
 - Automatic gain control, color balancing
 - On-chip lossy compression algorithms
 - Uncompressed images if desired
 - Integrated microphone, USB interface
 - Limitations
 - Narrow dynamic range
 - Narrow FOV (field of view)
 - Fixed spatial resolution
 - No actuation or active vision capabilities



Logitech, 2005

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

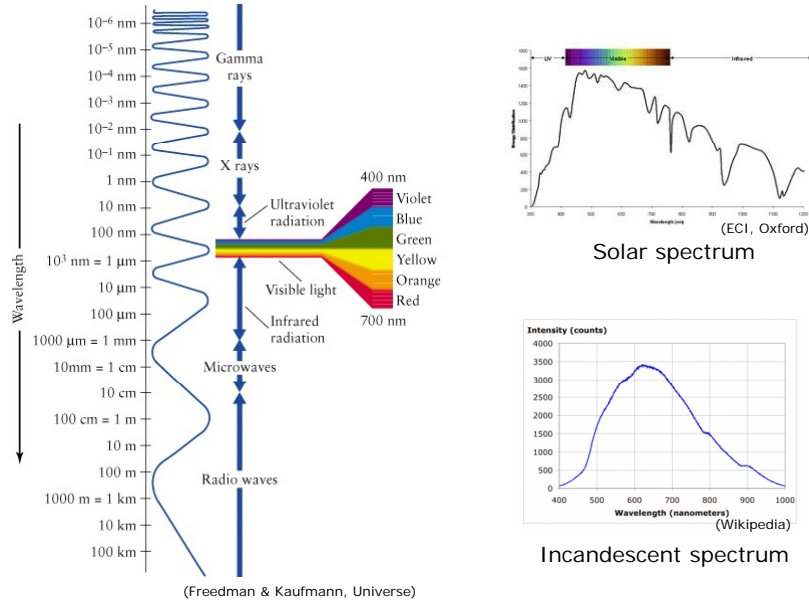


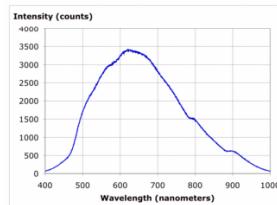
Image as measurement

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

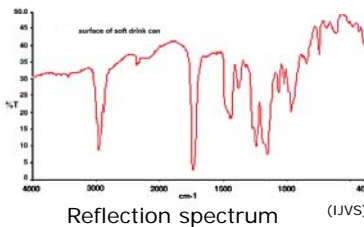


=

(at each image point)

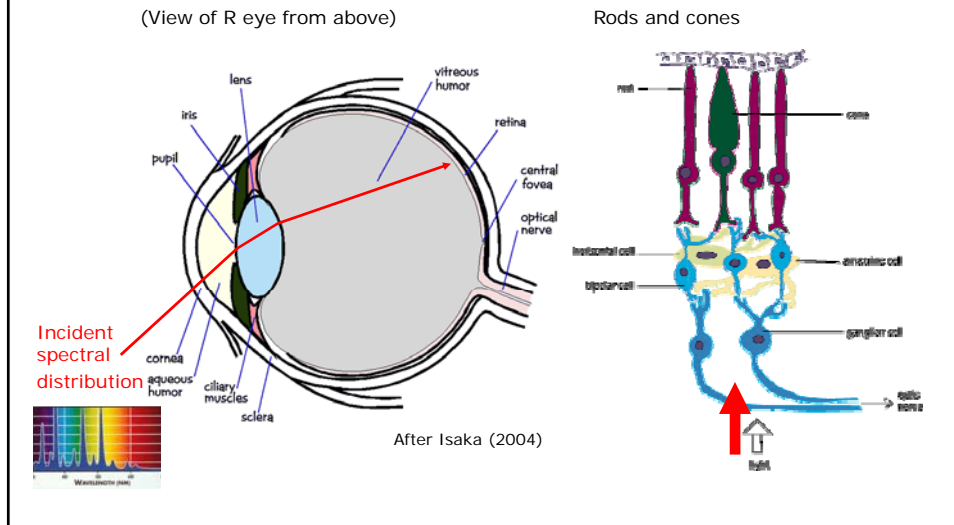


x



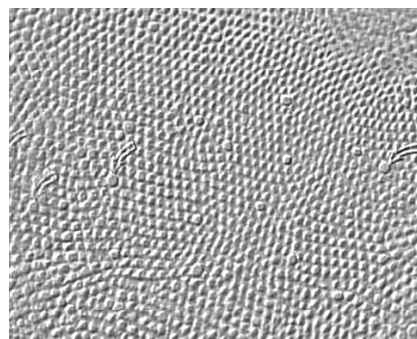
Human eye anatomy

- Spectrum incident on light-sensitive *retina*



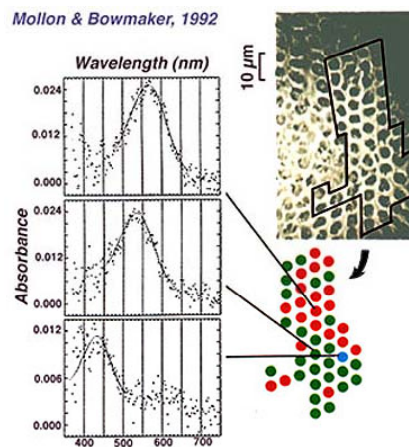
Foveal cone distribution

- Densely packed in fovea, less so in periphery



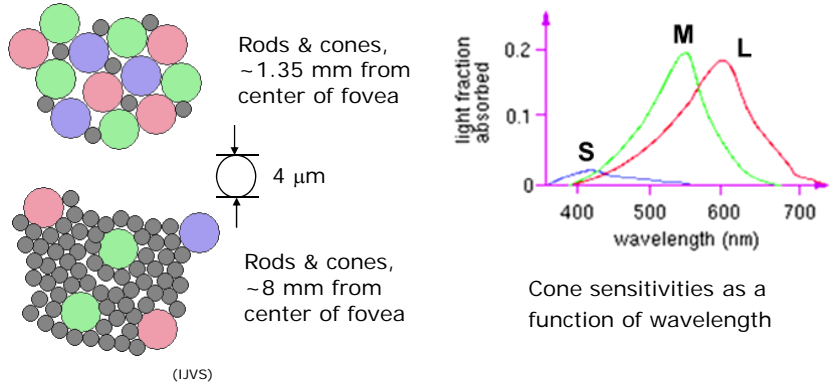
Visual discrimination of 1 minute of arc [corresponds roughly to] the center-to-center spacing ($3 \mu\text{m}$) of the cones of the central mosaics in the foveola (retina.umh.es).

What does "1 minute of arc" mean?



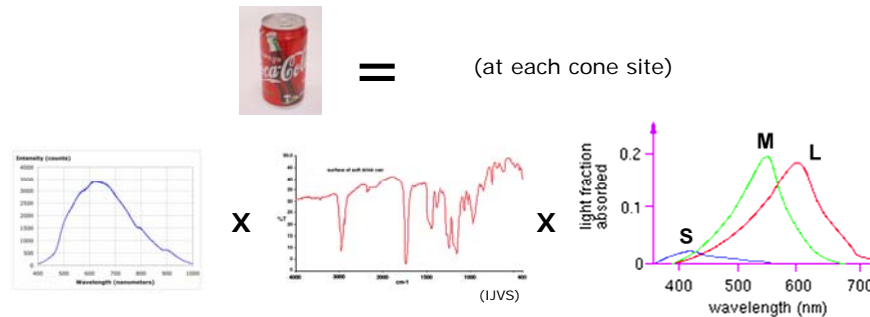
Cone sensitivities

- 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.



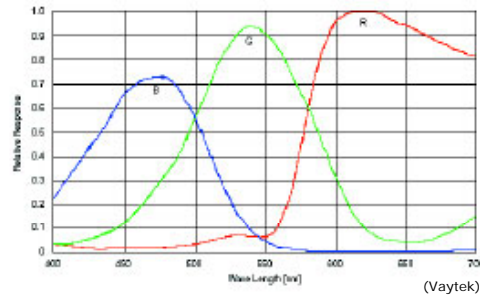
Color perception

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



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

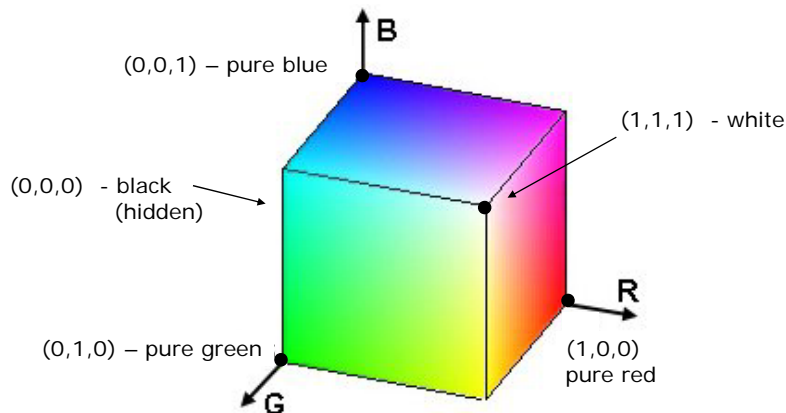


CCD with Bayer Filter, Relative Spectral Response Curve

- ... End of digression

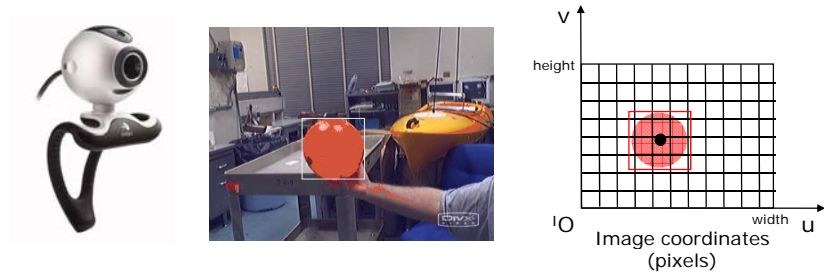
RGB Color Model

- Think of R, G, B as color orthobasis



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 the pixels of some desired color in the image ... right?

Naïve object detector

```
set objectPixels = ∅; // 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
```

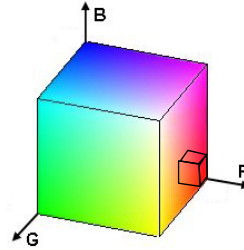
```
      objectPixels U= { (i, j) };
```

```
if ( isBall ( objectPixels ) ) // detector
```

```
  // do something in response to ball
```

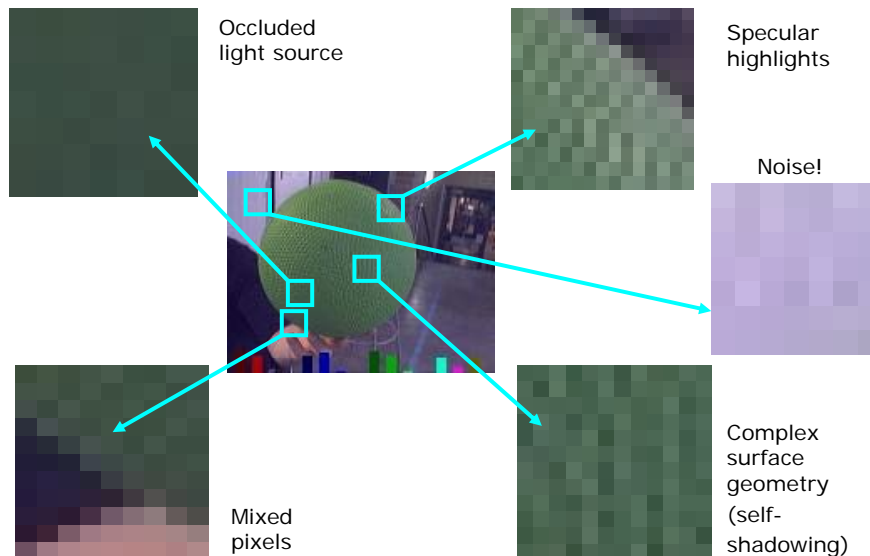
Pixel classification

```
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?

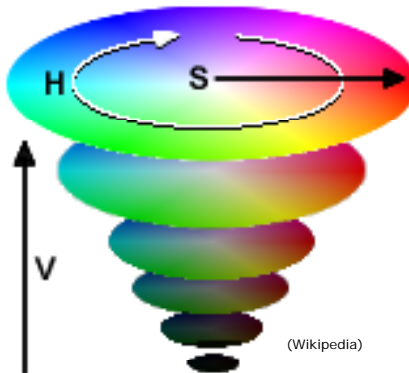
Real-world images



Human Visual System

- Adaptive over both short, long time scales
 - Brightness (dynamic range)
 - Color
 - Surround
- Variable resolution
 - Fovea
 - Periphery
 - Mix of color, intensity receptors
- Active
 - Saccading
 - Closed-loop eye tracking
 - Head and neck motions

HSV Color Model

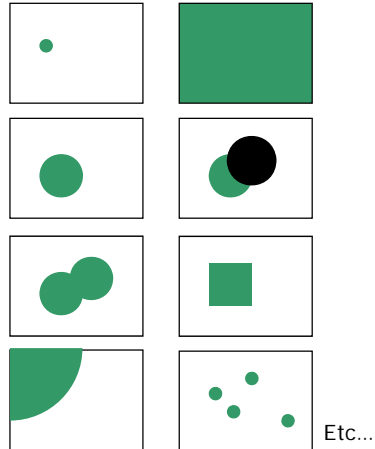


- More robust against illumination changes
- Still must confront noise, specularities etc.

Naive object detection

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

// how might this fail?



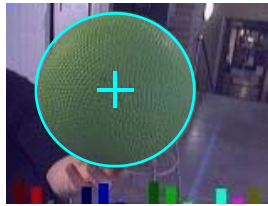
(Slightly) improved detection

```
Boolean isBall ( set s ) {  
  if ( |s| > 0.1 * W * H // big enough  
    && s is "ball-shaped" ) {  
    return true;  
  }  
  else  
    return false;  
}
```

// how might this fail?

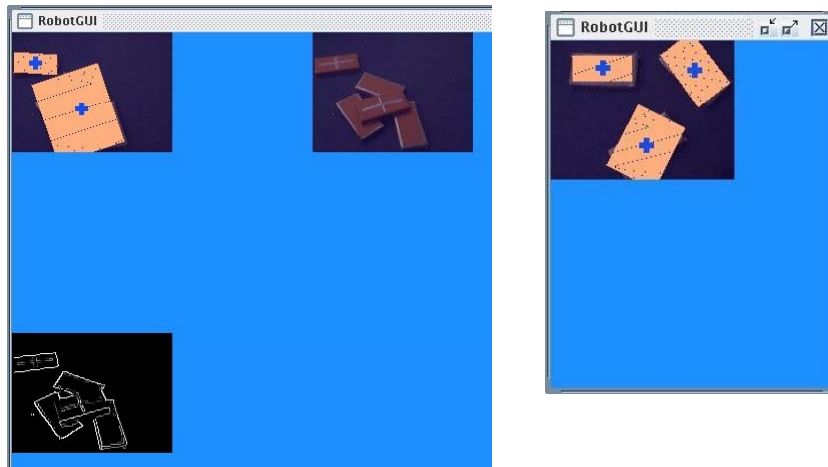
Doing something useful

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

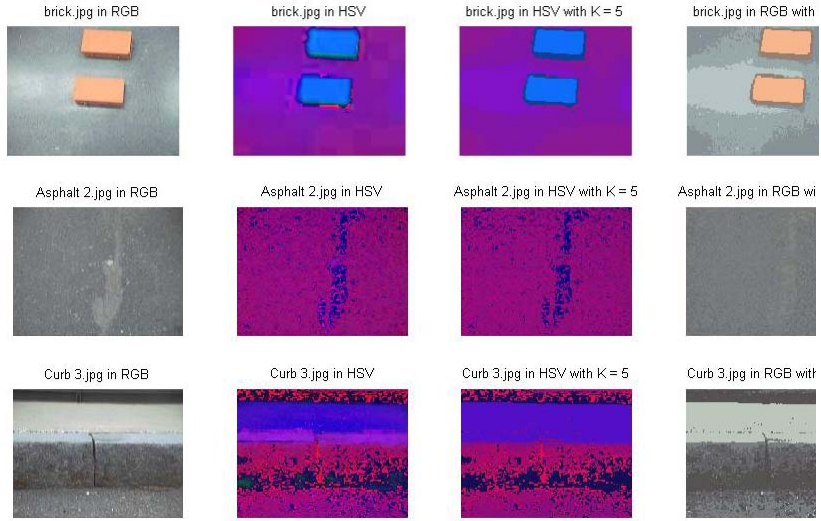


- How/when might these estimates be poor?

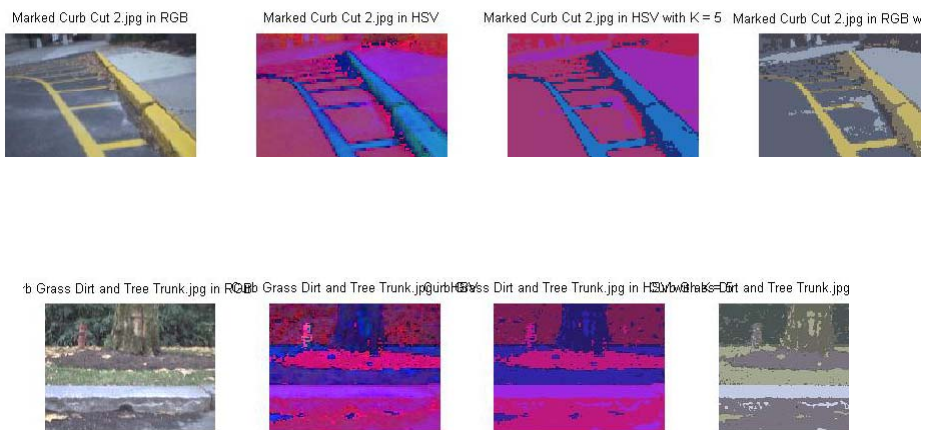
RSS student results



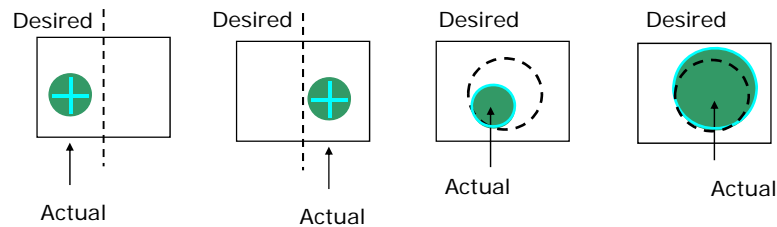
RSS student results



RSS student results



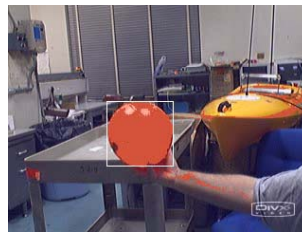
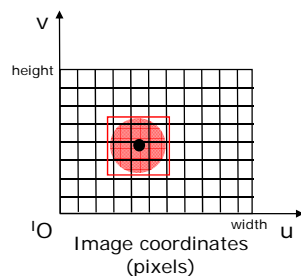
Size, centroid estimation



- Can use as inputs to motion controller!
- Suppose we want 1 m 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 5)

- Write “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 large connected components of pixels with that hue
 - Estimate the area, centroid of largest detected blob
- We will supply several “fiducial objects” (colored balls)
- Issue translation, rotation control so that robot “servos” to the ball: addresses it frontally, at desired standoff distance



Coming up in RSS:

- Today in Lab:
 - Lab 4 team presentations to faculty
 - Lab 5 (Visual servoing) begins
- Wednesday:
 - Lecture: Planning and Control (Prof. Rus)
 - Lab 5 continues
- Friday:
 - CI-M lecture on design reviews
 - ADD DATE
- Also this week:
 - First RSS “grades meeting”
(a third of the way through term)