Vision 2
15-491 CMRoboBits

Manuela Veloso
And
Brett Browning

Fall 2007

All images contained herein are either from the instructor(s) own work or publicly available on the web.

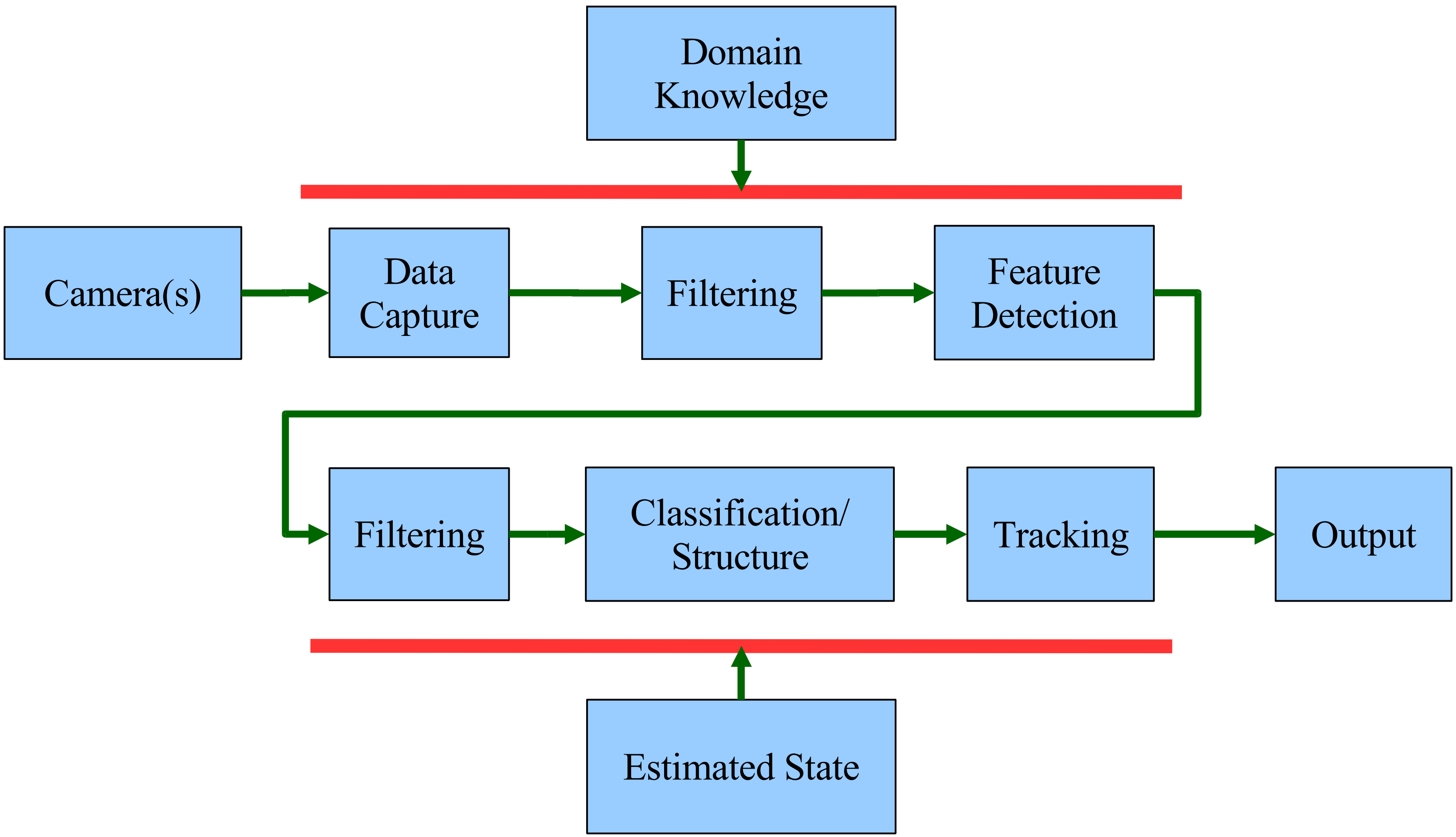
Outline

- Recap on Vision 1
- High level vision/perception ideas
- Simple object detection
- Tracking
- Summary

Vision Algorithms Overview

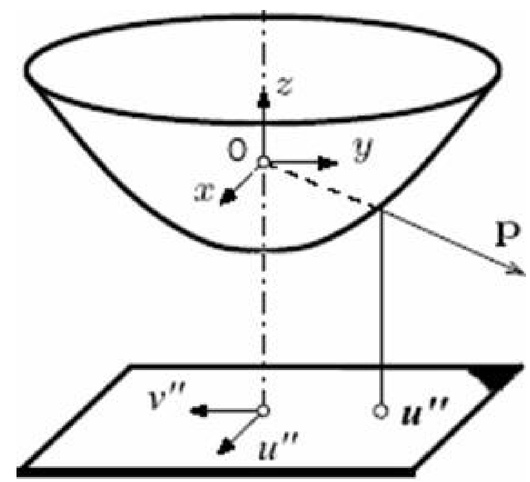
- For robots, vision used for two key problems
- Finding objects
  - Object detection, recognition, and tracking
- Understanding structure
  - Structure from motion/stereo
  - SLAM
  - Structure/shape from texture

Typical Parts of a Vision System

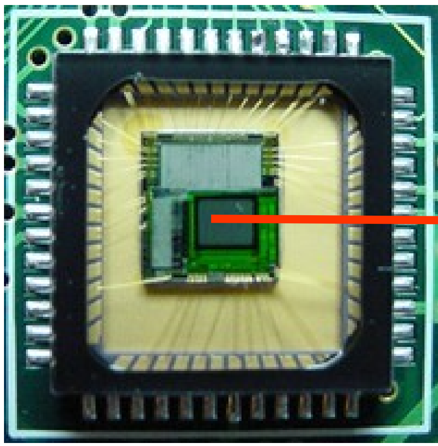


Cameras As Sensors

- Most machine vision cameras consist of
  - Photon sensitive sensor elements with filters
  - Mirror(s) and/or lens(es) to manipulate light
  - Digital frame capture electronics
  - Optionally structured light sources



Parts of a Digital Camera



CMOS Image sensor from “how stuff works”

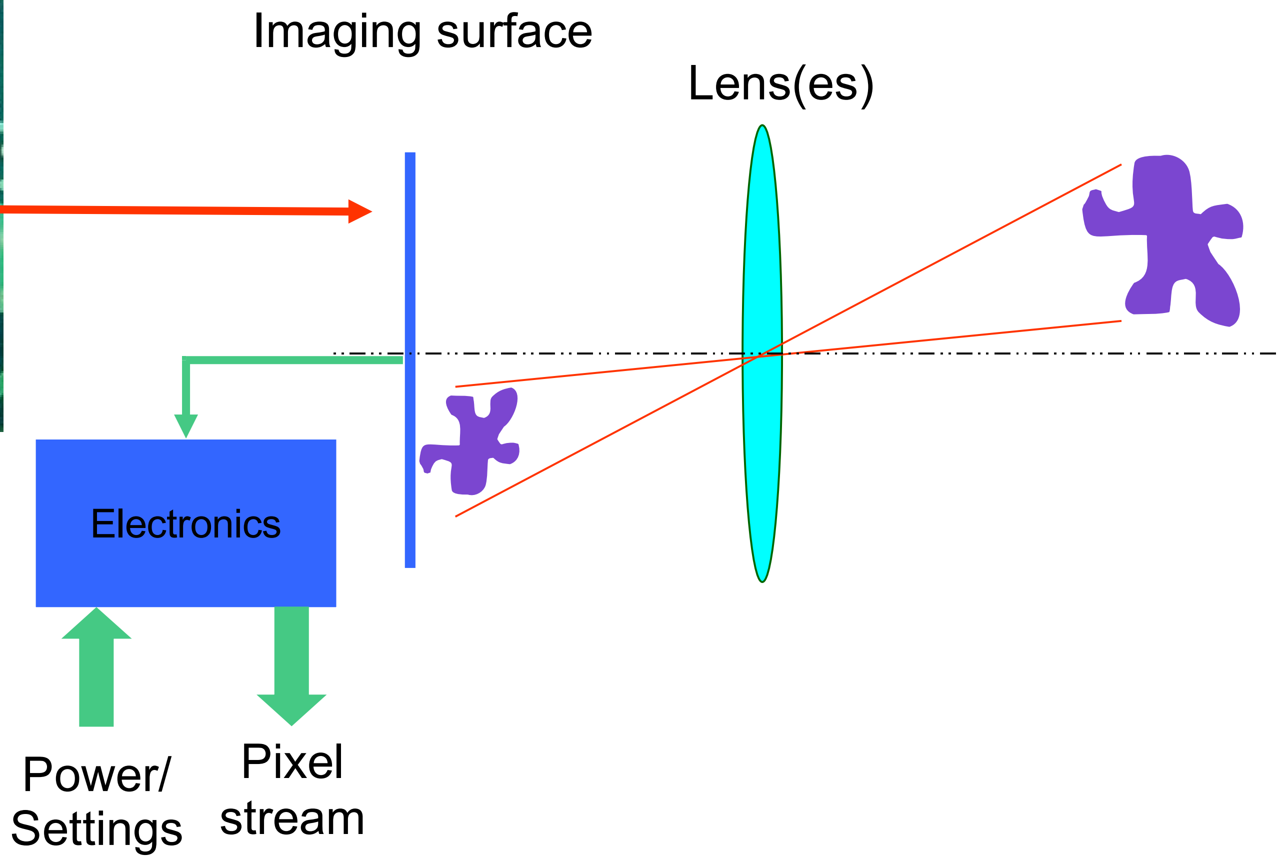
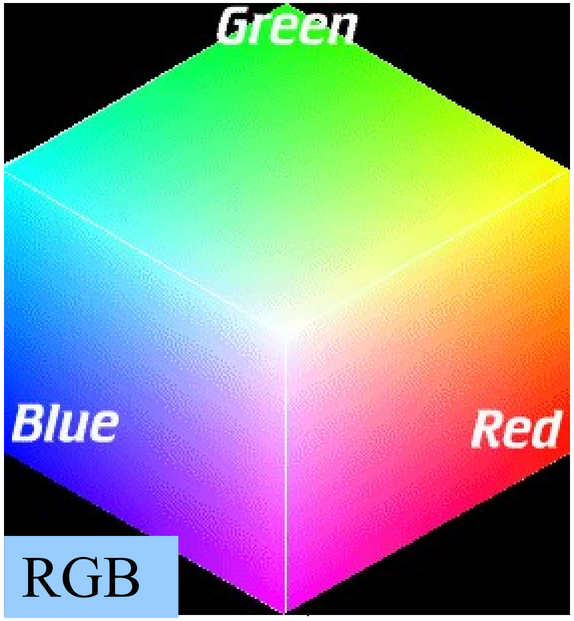


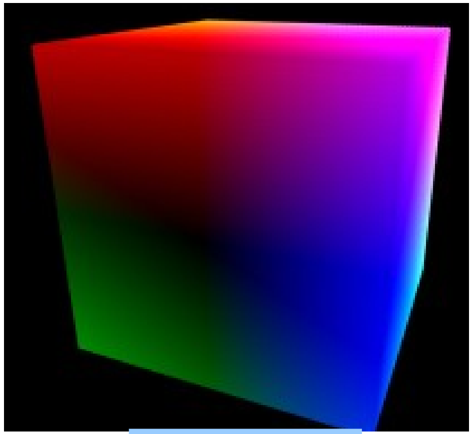
Image Example

Color Spaces

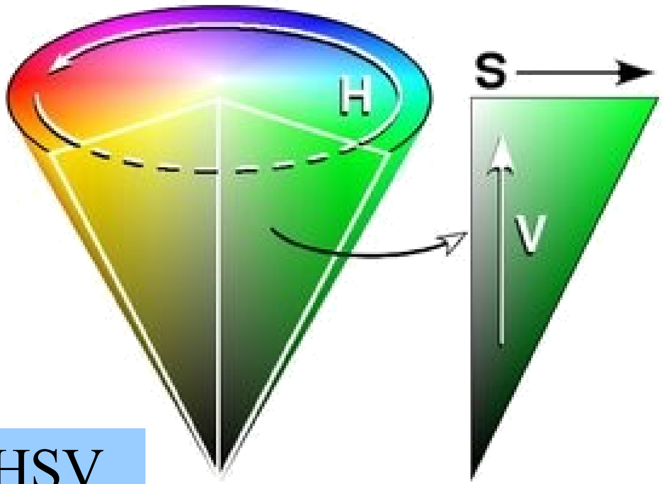
- Many ways to represent color
  - RGB (red, green, blue), nRGB
  - YUV, or Y Cr Cb (luminance + chroma)
  - HSV or HSL (hue, saturation, value)



RGB



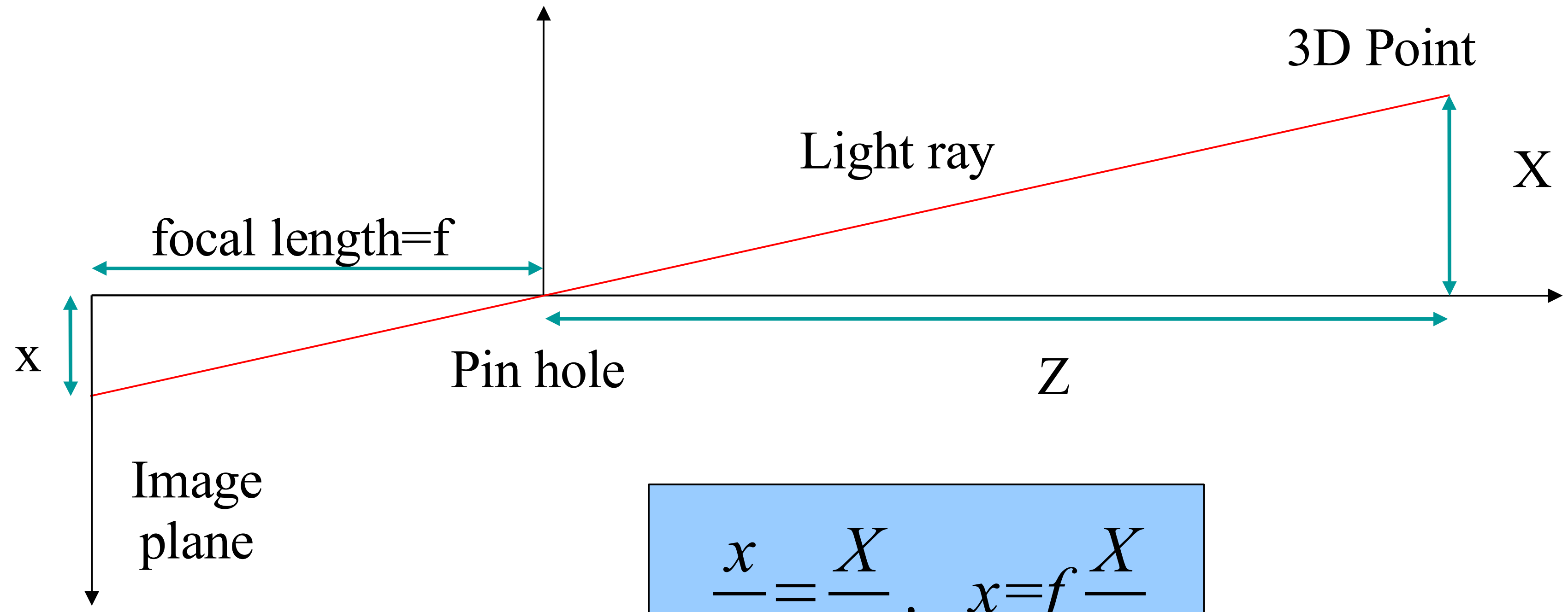
YUV



HSV

2D Pin-hole Camera

- Using similar triangles



$$\frac{x}{f} = \frac{X}{Z}, \quad x = f\frac{X}{Z}$$

All Together

Extrinsic parameters

$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \propto KR\left(X - X_0\right)$$

Intrinsic parameters

$$K = \begin{pmatrix} f\alpha_x & 0 & c_x \\ 0 & f\alpha_y & c_y \\ 0 & 0 & 1 \end{pmatrix}$$

Convolution Example

- Operator is a matrix
- Result is another image

-1	0	1
-2	0	2
-1	0	1

Operator Window

Image

3	4	20	21	20	19	0	1
3	2	3	20	20	19	2	1
2	0	3	19	21	20	3	0
2	1	1	25	24	19	21	5
5	10	19	20	23	3	2	0
1	3	10	20	19	24	1	1

Apply operator at a location in the image

Fast Color Segmentation

- Classify each pixel based on color using pre-defined tables, then group pixels into blobs

Symbolic color value: {Unknown, Red, Yellow, Blue, Green, Floor}

High Level Vision

High Level Vision Goals

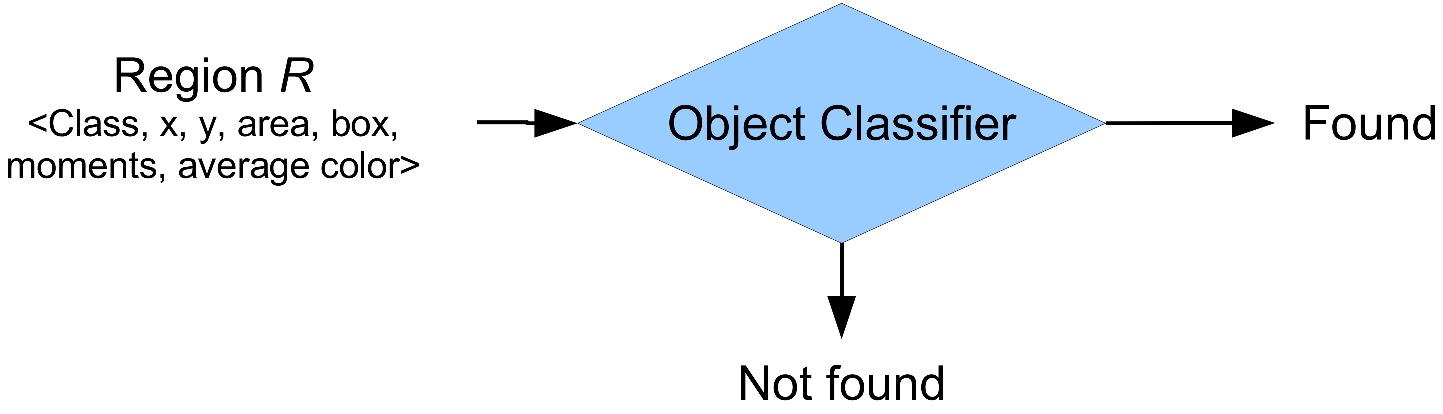
- Populate world model for robot
- Output is the input for behaviors and/or planning
- Key challenges
  - Signal to noise ratio
  - Tracking and data association
  - Inferring non-observable information
  - Estimating and using confidence bounds

Two Types of Output

- Objects
  - Pose, motion, articulation, type
  - Result of object detection and tracking
- World structure
  - 2D occupancy grids, 2.5D/3D maps
  - Textured surfaces, point clouds
  - Vehicle pose/trajectory

Simple Object Detection

- Single colored object
  – Low-level vision produces *regions*
  – Look for *region* of right size/shape/color
- This is pattern recognition or classification!



Classification Recap

- Supervised learning
- Input is *x* output is *y*
  - Classification: y={-1,1}
- Given labeled training examples
  - $\{(x_1,y_1),(x_2,y_2), \ldots,(x_N,y_N)\}$
  - Learn a classifier: f(x) that minimizes loss function L(f(x),y) over data e.g. sum squared error

Classifiers

- Many possibilities
  - Support Vector Machines (e.g. libsvm)
  - Neural networks, Decision trees (e.g. C4.5)
  - Naïve Bayes, Nearest neighbor, ...
- Can use dimensionality reduction
  - PCA, kernel-PCA
- Can use quantization methods
  - LVQ, K-means

An Example

- …

Important Points

- Data collection
  - Needs training data distribution to match expected real distributions
  - Training should include range of lighting, pose, conditions etc.
- Detection needs to be fast for real-time application

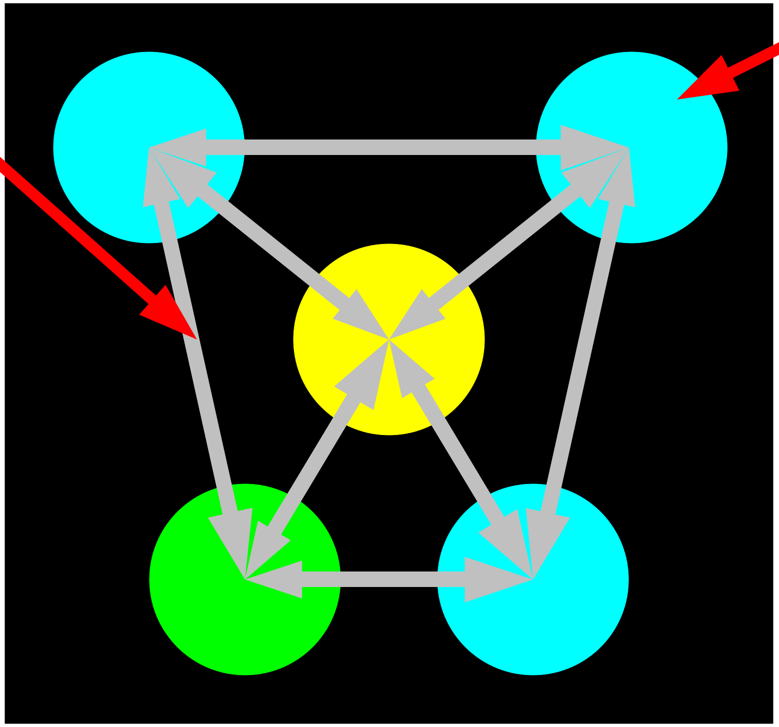
Multiple Components

- Single region is a bit limiting...would like to have multi-region objects
- More complex recognition process
  - Each "feature" has particular properties
  - "Features" have geometrical relationship
- Recognition issues
  - How to recognize "parts"
  - How to recognize whole from parts
  - Which comes first?

Multi-part Objects

Match Geometry

Detect as before



Matching Geometry

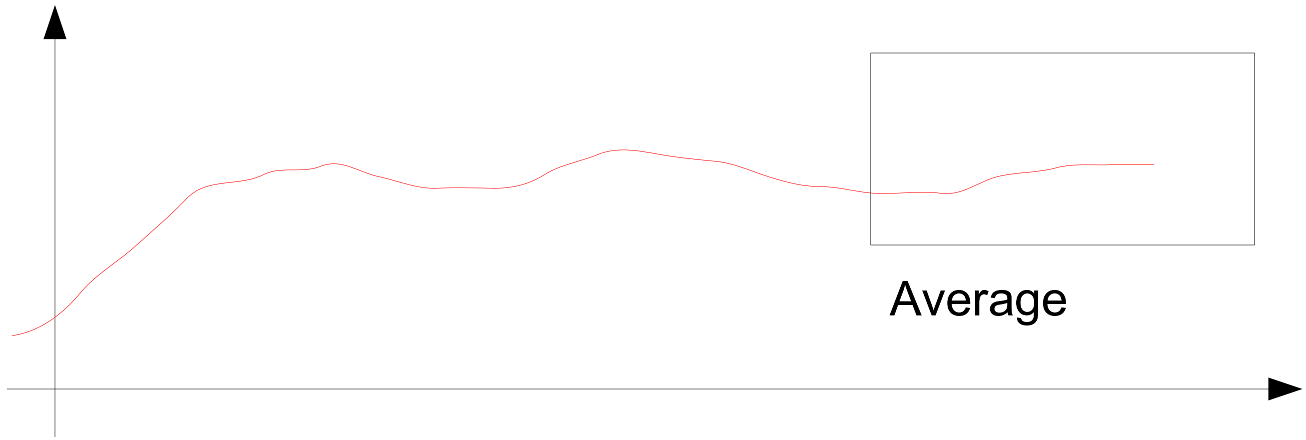
- Need to account for
  - Distortions
  - Missed detections
- Hough-style approach
  - Voting on model configuration
- Graphical model style approach
  - Hidden Markov Model
- Ransac style approach
  - Match based on simple transform: Euclidean, Affine, Perspective

Tracking

- The need
  - May not detect object in every frame
  - Direct estimates may be noisy
  - Unobserved parameters (e.g. speed)
- Approaches
  - Simple linear filters
  - Kalman filters

Simple Filtering

- Take a locally weighted average of recent observations



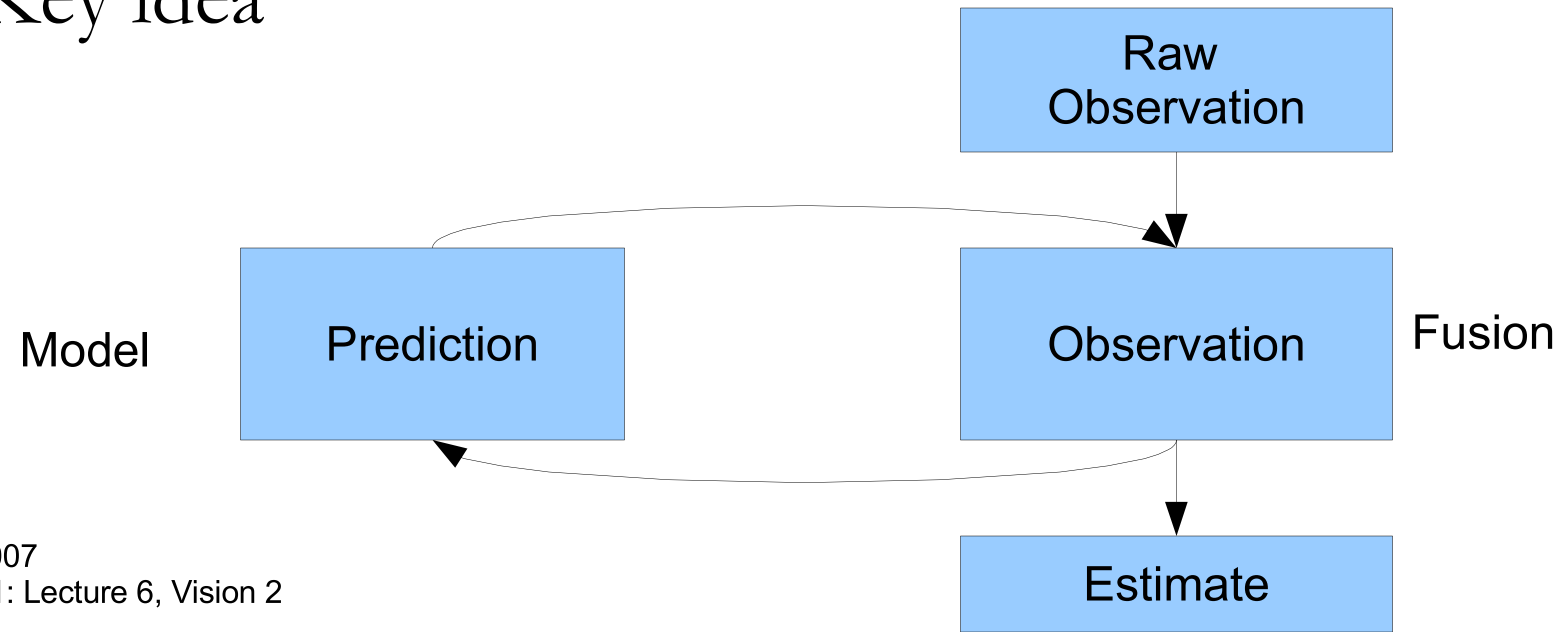
- Could weight averages
  - FIR: Finite Impulse Response Filter

Pros/Cons

- Advantages
  - Very simple, easy to implement
  - Will not "over-estimate" (bounded error)
  - Can estimate confidence by local statistics
    - e.g. variance in window
- Disadvantages
  - Latency a function of window size
  - Doesn't give extra parameters
    - e.g. speed

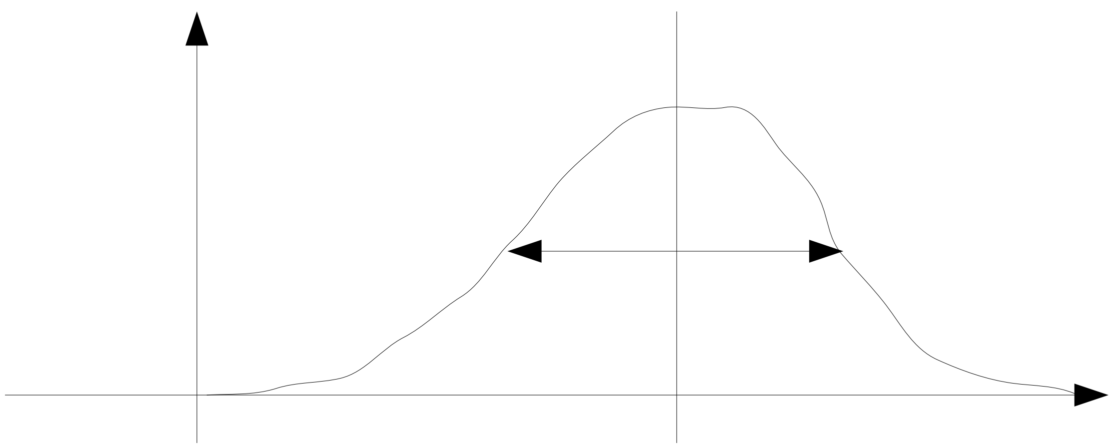
Can We Do Better?

- Yes, if we have a model of the system
- Enter Kalman filtering...
  - Reference: http://www.cs.unc.edu/~tracker/ref/s2001/kalman/ir
- Key idea



Basic Idea

- Signals have noise so estimate is uncertain
  - Model uncertainty as a Gaussian
- So our belief model is
  - x – mean, P – covariance



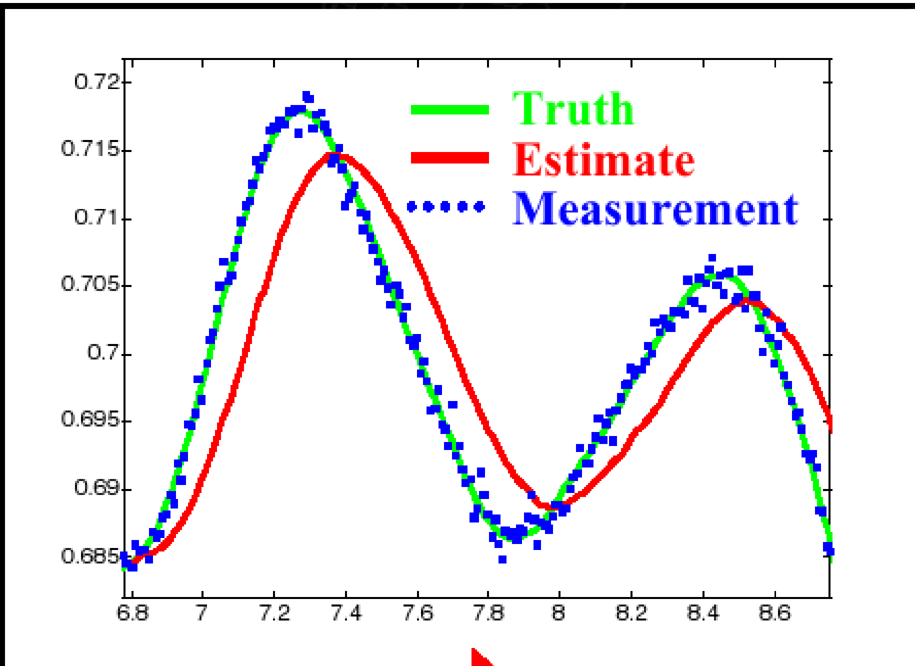
Model Part

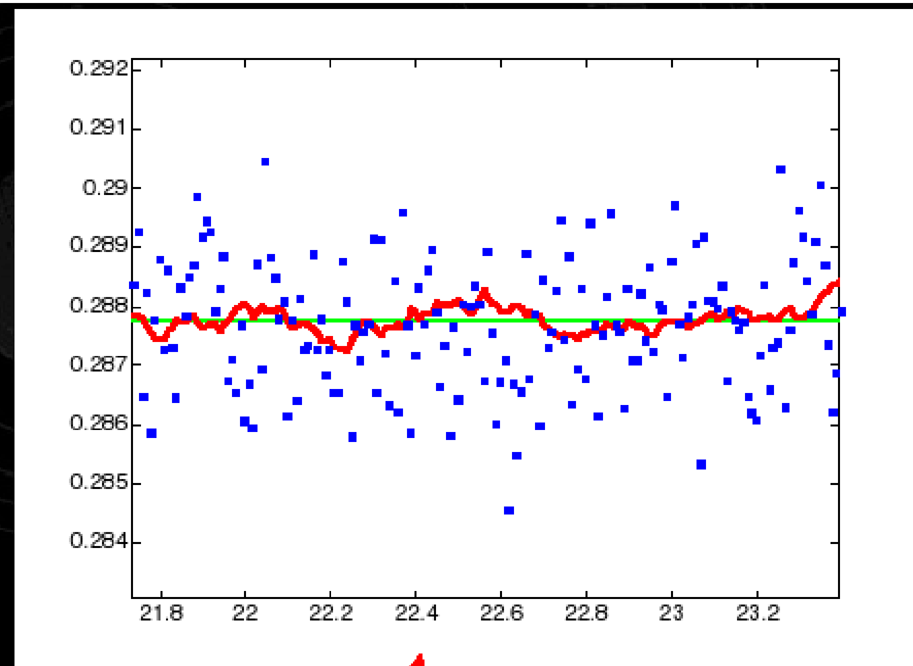
- We can update the mean by using a model of the system f(x)=Ax+b
- How do we update covariances?
  - Recall Var(Ax+b)=A' Var(x) A
- Using these we can predict where the target will be, and with what confidence

Observation Part

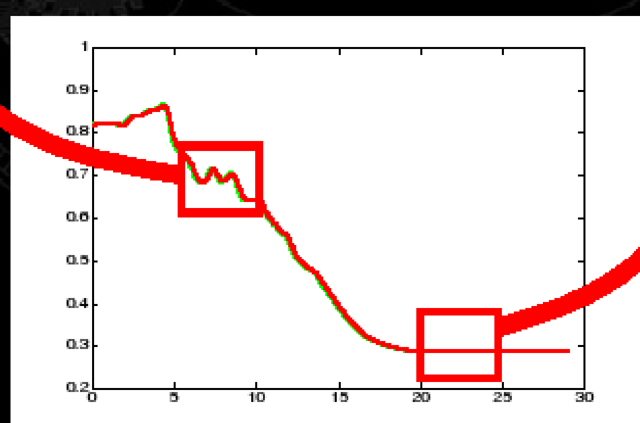
- Define z: observation
- We want $P(x'|x,z)$
- It's Gaussian, so this is easy

Example from Welch et al.





significant *latency* when moving…



…relatively *smooth* when not

Questions?