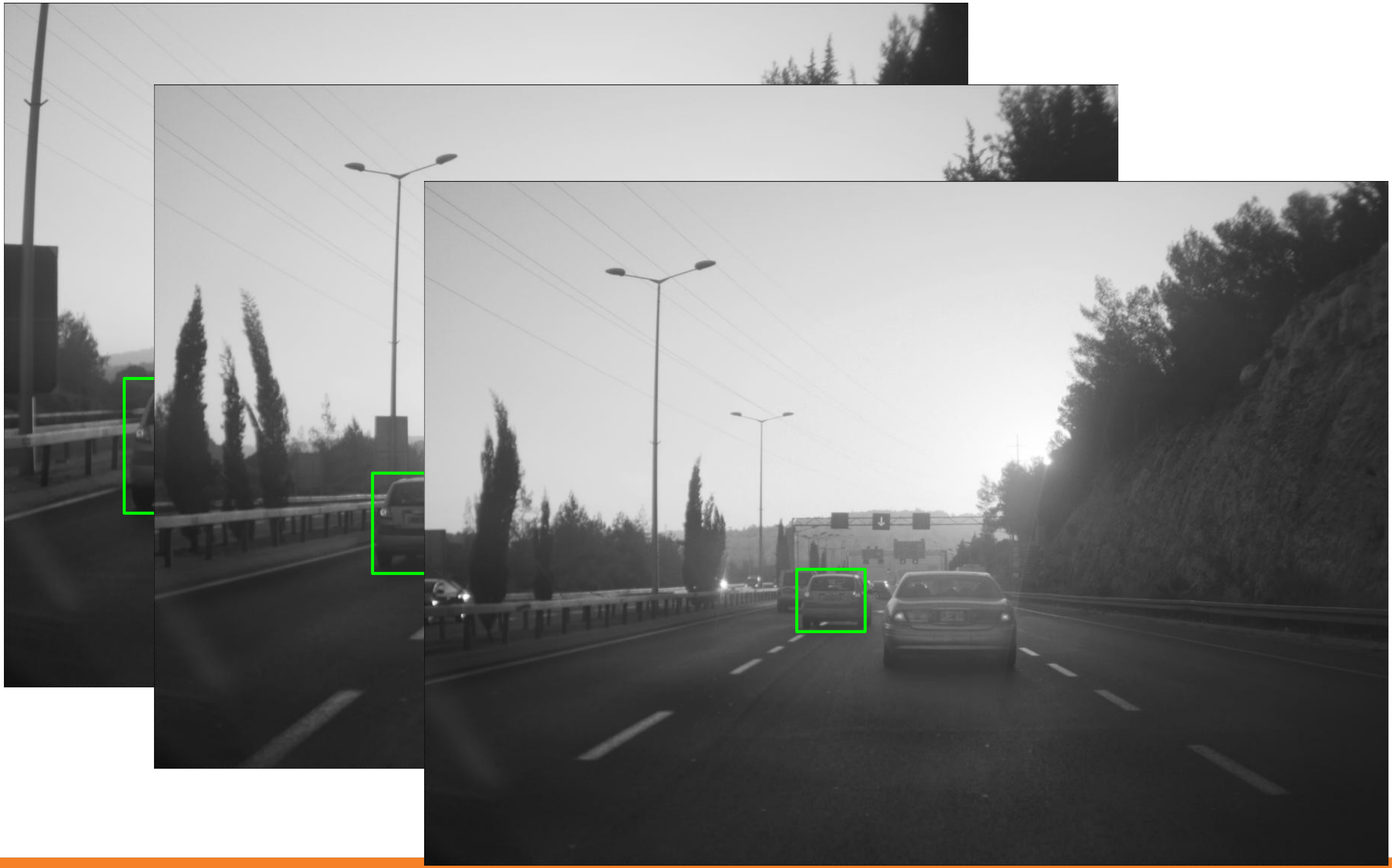
Lecture 13
Tracking

COS 429: Computer Vision

PRINCETON UNIVERSITY

Slides credit:
Many slides adapted from James Hays, Derek Hoeim, Lana Lazebnik, Silvio Saverse, who in turn adapted slides from Steve Seitz, Rick Szeliski, Martial Hebert, Mark Pollefeys, and others

The goal



Slide Credit:

Tracking

Object Tracking: Learn some representation of the object, and use that representation to find it in subsequent frames

Future Image Frame

Template

Slide Credit:

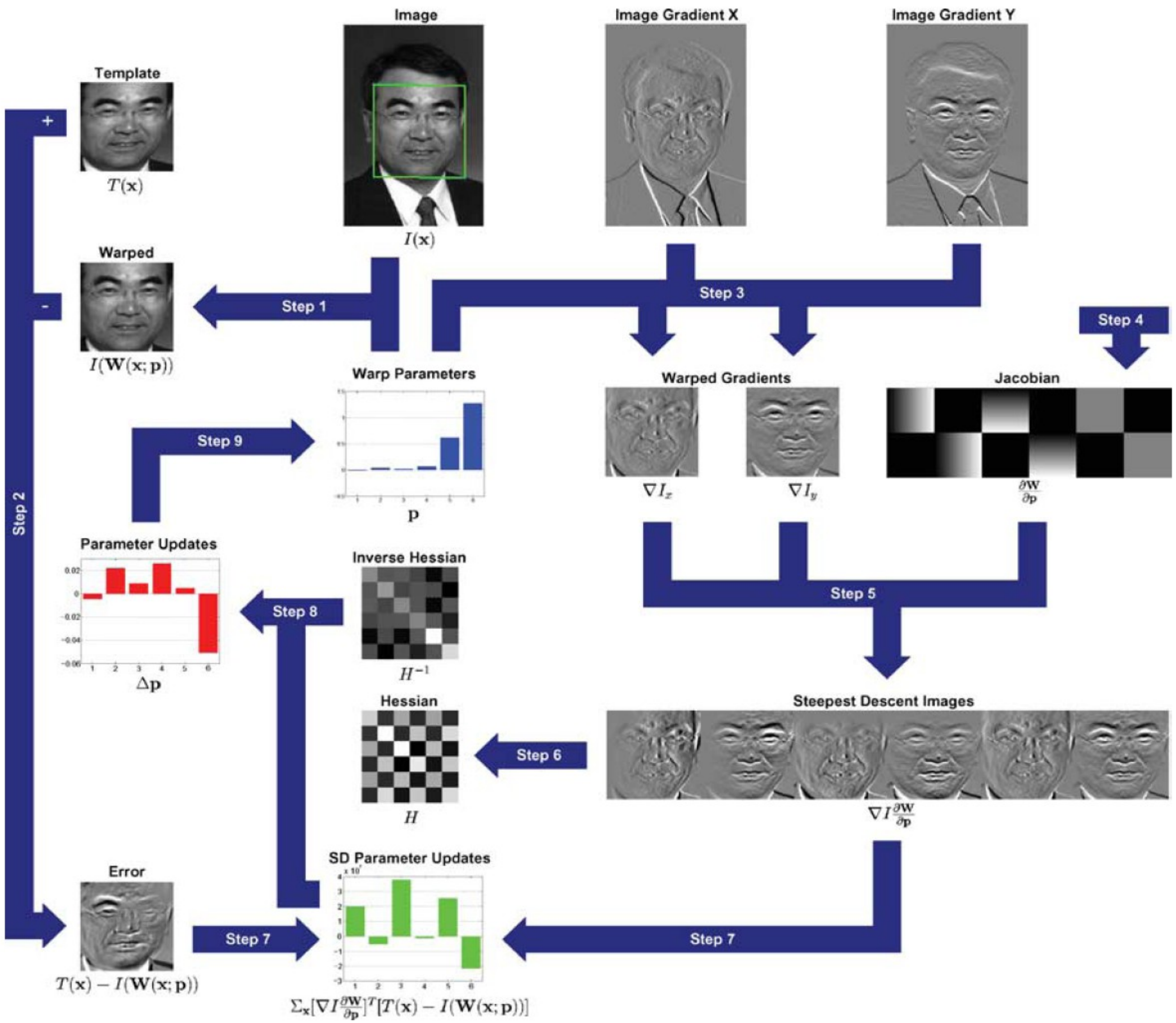
Approaches to Object Tracking

- Motion model (translation, translation+scale, affine, non-rigid, …)
- Image representation (gray/color pixel, edge image, histogram, HOG, wavelet...)
- Distance metric (L1, L2, normalized correlation, Chi-Squared, …)
- Method of optimization (gradient descent, naive search, combinatoric search...)
- What is tracked: whole object or selected features

Template

Slide Credit:

Schematic of Lucas-Kanade

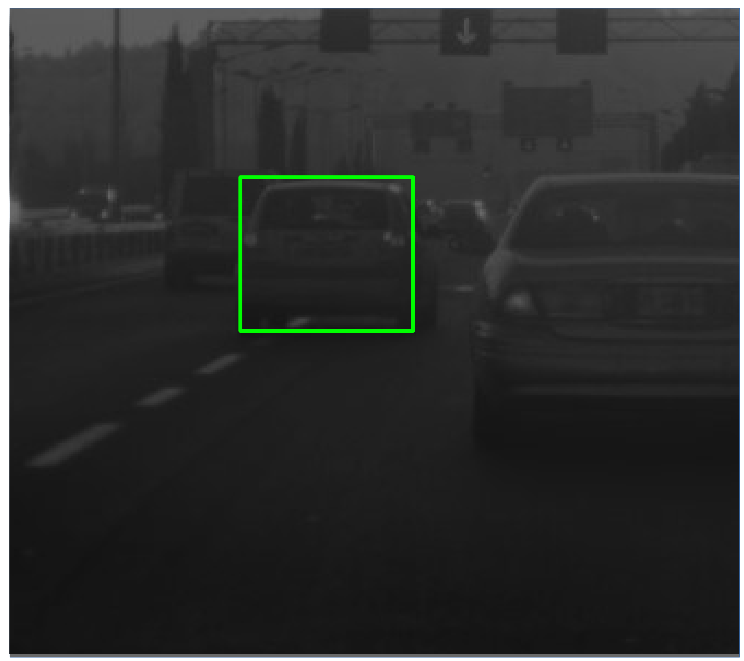


 Slide Credit:[Baker & Matthews, 2003]

LK Problem: Change in Brightness

Possible Solutions:

- Subtract mean intensity (based on current estimate before iteration)
- Transform gray values into some features that are not effected by brightness
  - Any filter that is zero-mean
  - Example: vertical, horizontal edge filters
  - Example: Non-parametric filters (Rank, Census Transforms)



Slide Credit:

More Problems

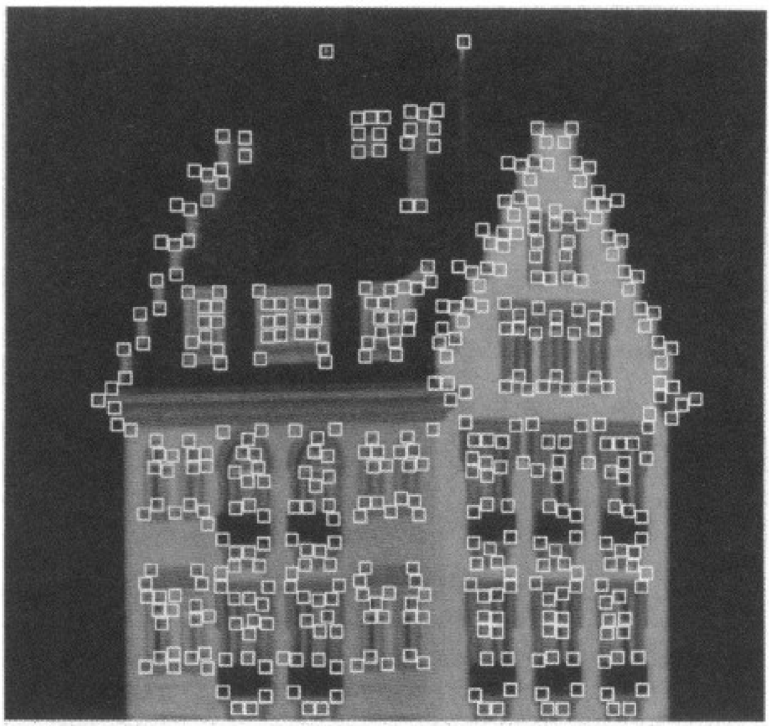
- Outliers: bright strong features that are wrong

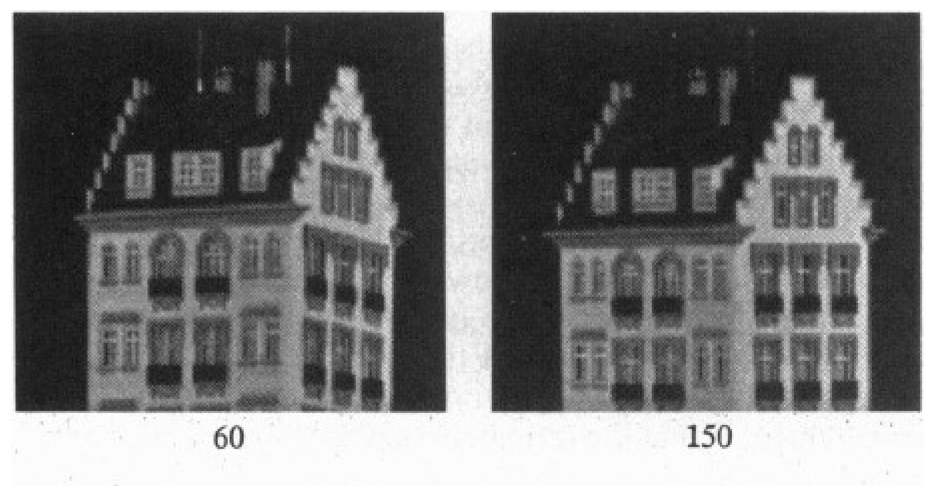
- Complex, high dimensional, or non-rigid motion

Slide Credit:

Feature Tracking

- Similar to feature matching, but track instead of match:
  - Track small, good features using translation only (u,v)
  - Use RANSAC to solve more complex motion model (Scale, Rotation, Similarity, Affine, Homography, ... Articulated, non-rigid)



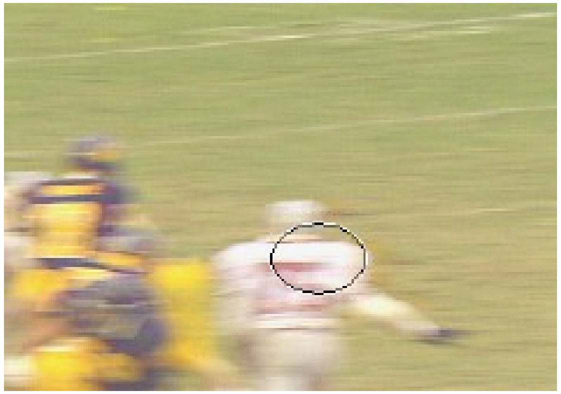


Can any of these techniques handle this?

Template

Partial occlusion

Distraction

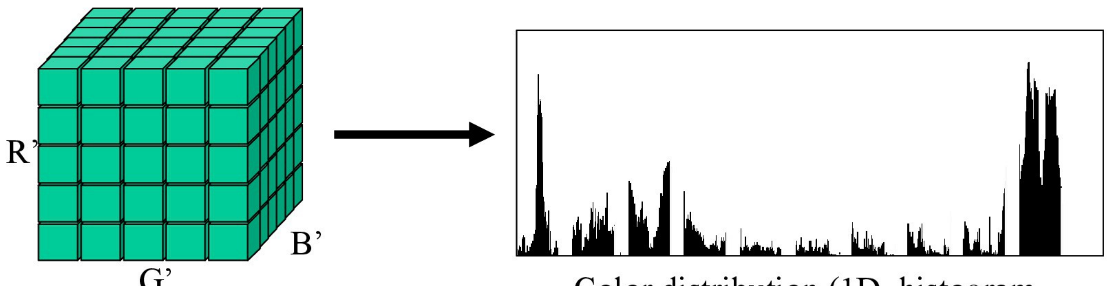


Motion blur

Slide Credit:[Baker & Matthews, 2003]

Robert Collins
CSE598C, PSU

Appearance via Color Histograms



discretize

Color distribution (1D histogram normalized to have unit weight)

R' = R << (8 - nbits)
G' = G << (8 - nbits)
B' = B << (8-nbits)

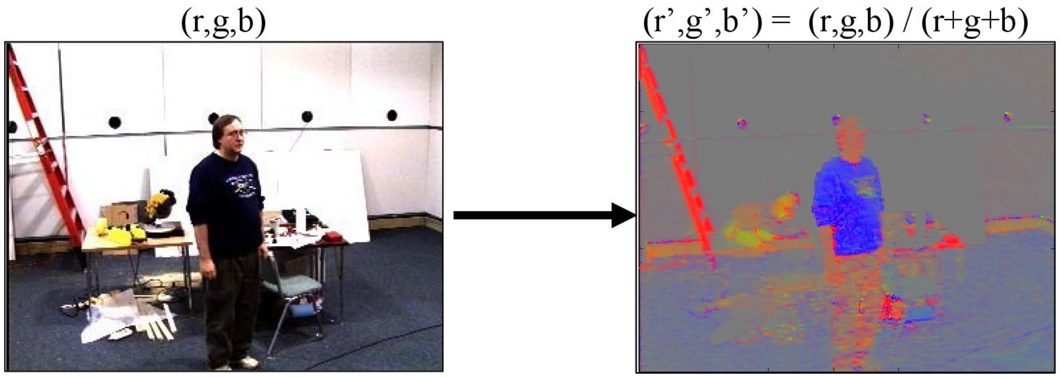
Total histogram size is (2^(8-nbits))^3

example, 4-bit encoding of R,G and B channels yields a histogram of size 16*16*16 = 4096.

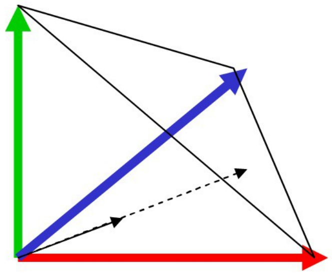
Slide Credit:

Robert Collins
CSE598C, PSU

Normalized Color



Normalized color divides out pixel luminance (brightness), leaving behind only chromaticity (color) information. The result is less sensitive to variations due to illumination/shading.



Slide Credit:

Comparing Color Distributions

Bhattacharya Distance:

Given an n-bucket model histogram $\{m_i \mid i=1,\ldots,n\}$ and data histogram $\{d_i \mid i=1,\ldots,n\}$, we follow Comanesciu, Ramesh and Meer * to use the distance function:

$$\Delta(m,d) = \sqrt{1 - \sum_{i=1}^{n} \sqrt{m_i \times d_i}}$$

Similarity Function

$$f(y) = f\left[\vec{p}(y), \vec{q}\right]$$

Why?

1) it shares optimality properties with the notion of Bayes error
2) it imposes a metric structure
3) it is relatively invariant to object size (number of pixels)
4) it is valid for arbitrary distributions (not just Gaussian ones)

*Dorin Comanesciu, V. Ramesh and Peter Meer, "Real-time Tracking of Non-Rigid Objects using Mean Shift," IEEE Conference on Computer Vision and Pattern Recognition, Hilton Head, South Carolina, 2000 (best paper award).

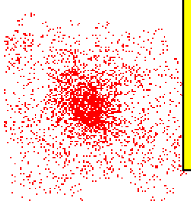
Slide Credit:

How to optimize histogram agreement?

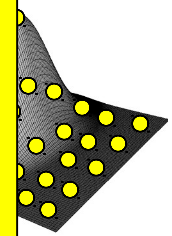
Recall: Mean Shift
Finding modes in a set of data samples, manifesting an underlying probability density function (PDF) in R^N

<u>PDF in feature space</u>

- **Color space**
- **Scale space**
- **Actually any feature space you can conceive**
- **…**

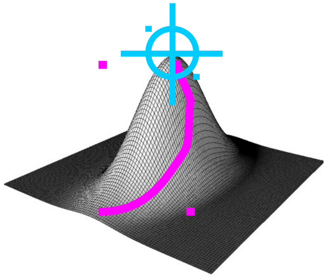


Data



PDF Representation

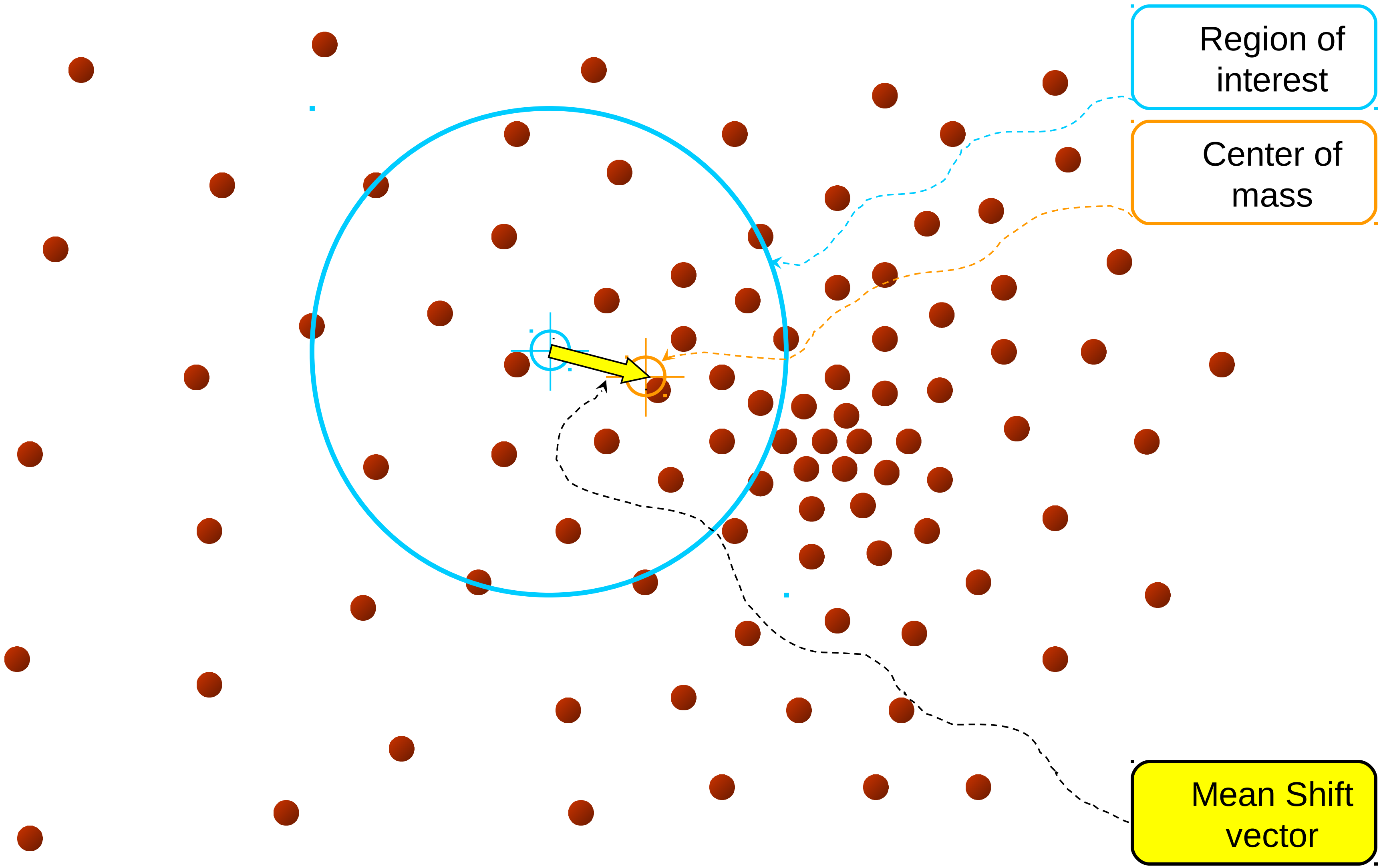
Non-parametric
Density **GRADIENT** Estimation
(Mean Shift)



PDF Analysis

Slide Credit: **Y. Ukrainitz & B. Sarel**

Intuitive Description

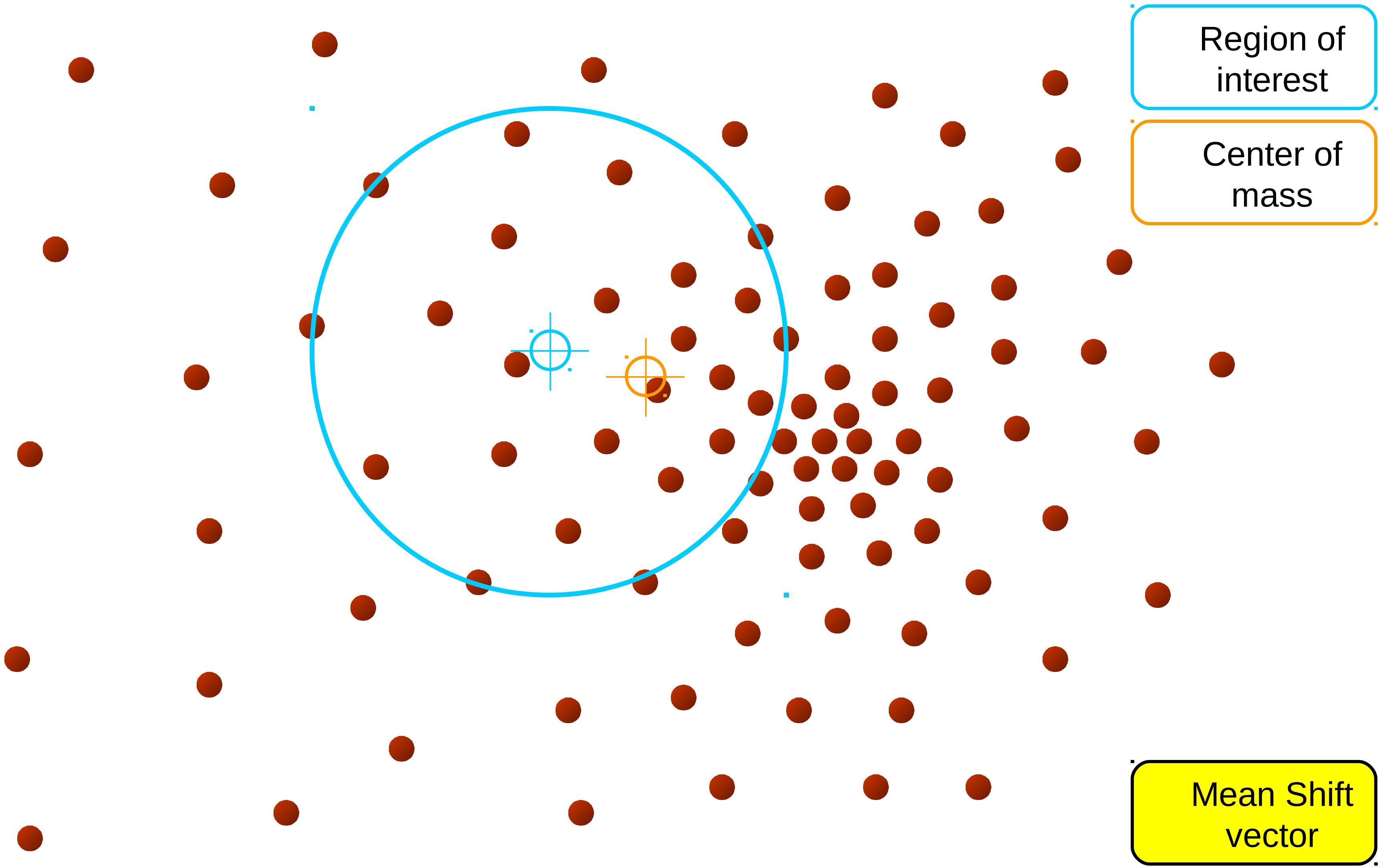


Objective : Find the densest region

Distribution of identical billiard balls

 Slide Credit: **Y. Ukrainitz & B. Sarel**

Intuitive Description

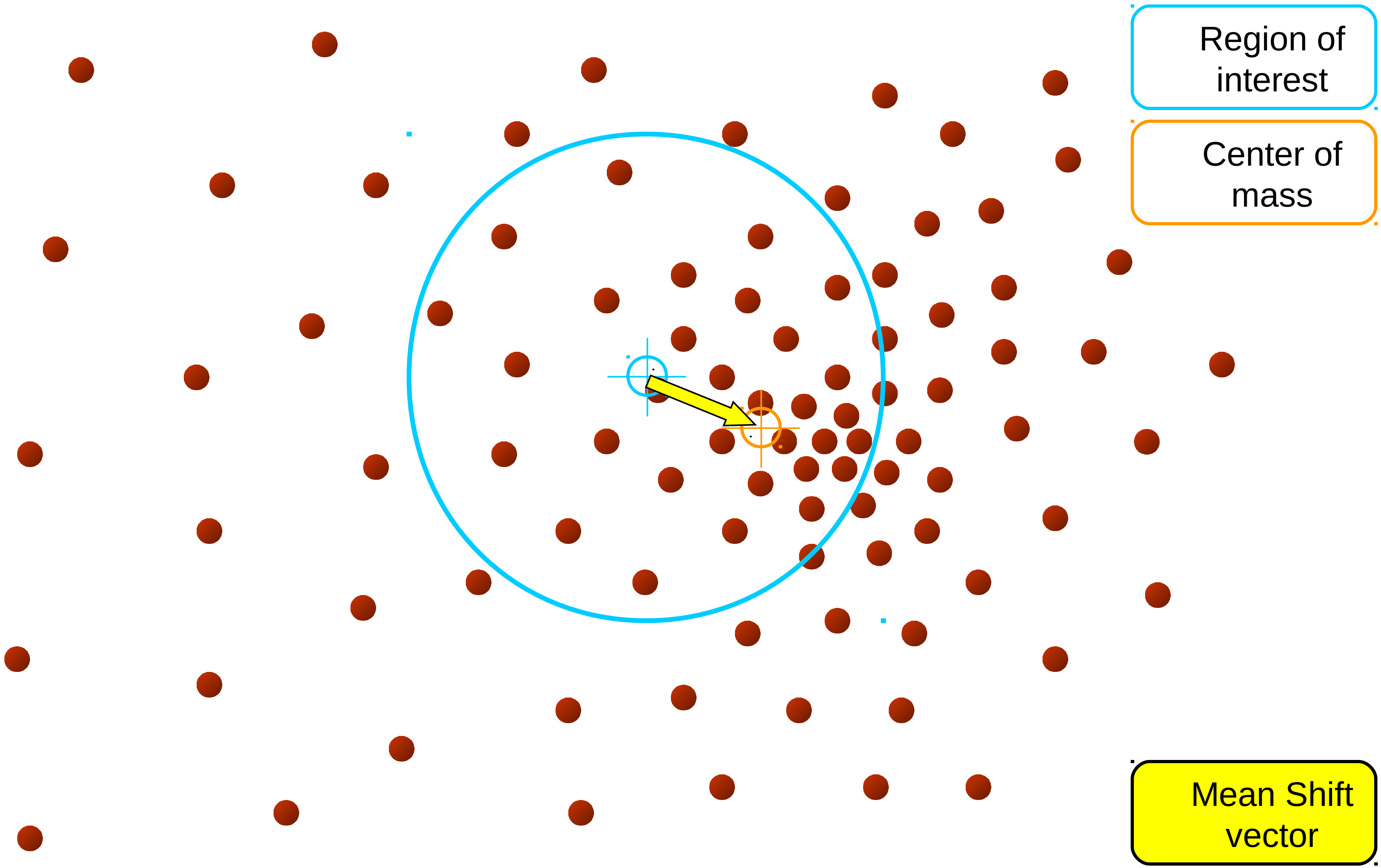


Objective : Find the densest region

Distribution of identical billiard balls

Slide Credit: **Y. Ukrainitz & B. Sarel**

Intuitive Description

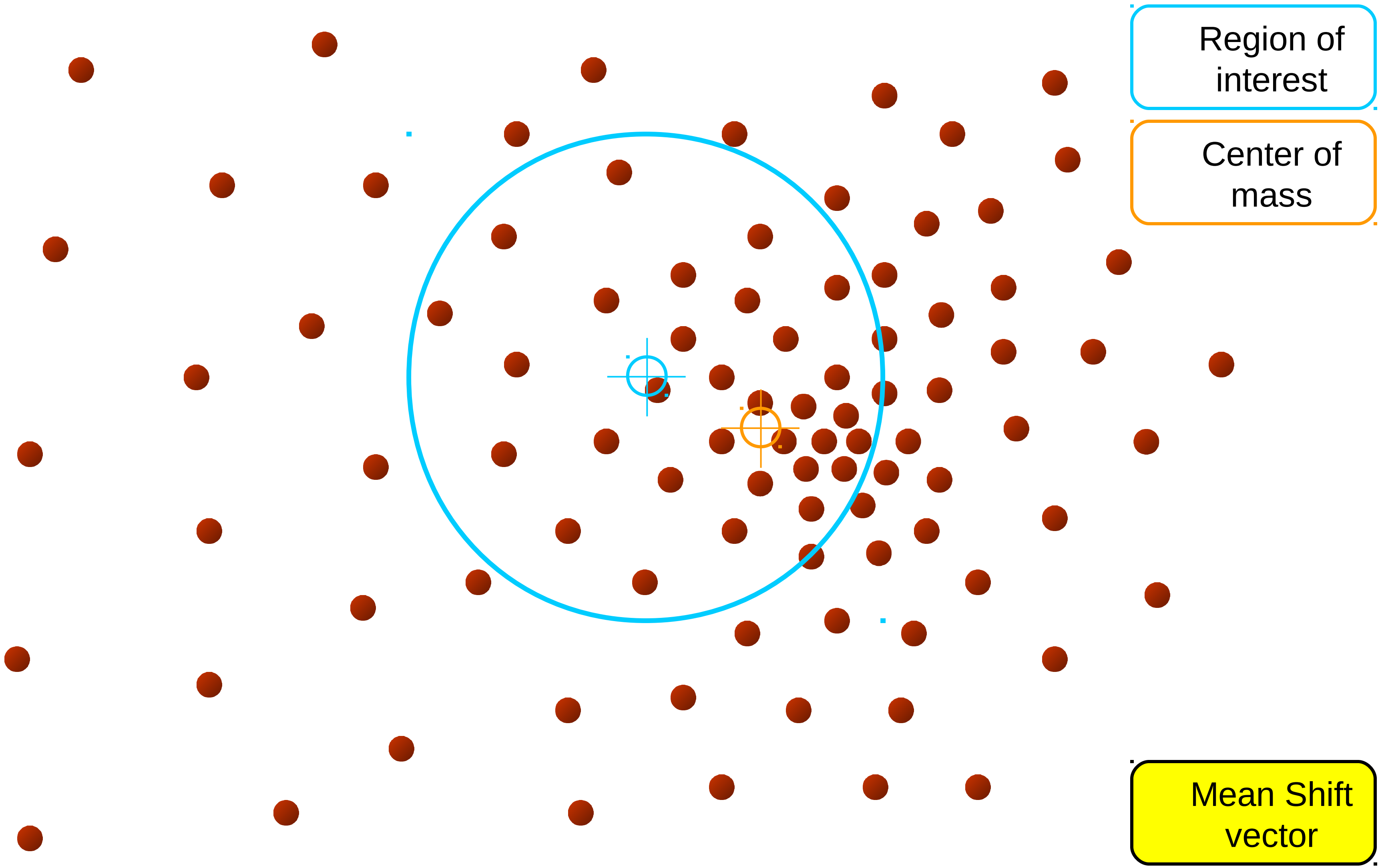


<u>Objective </u>: Find the densest region

Distribution of identical billiard balls

 Slide Credit: **Y. Ukrainitz & B. Sarel**

Intuitive Description

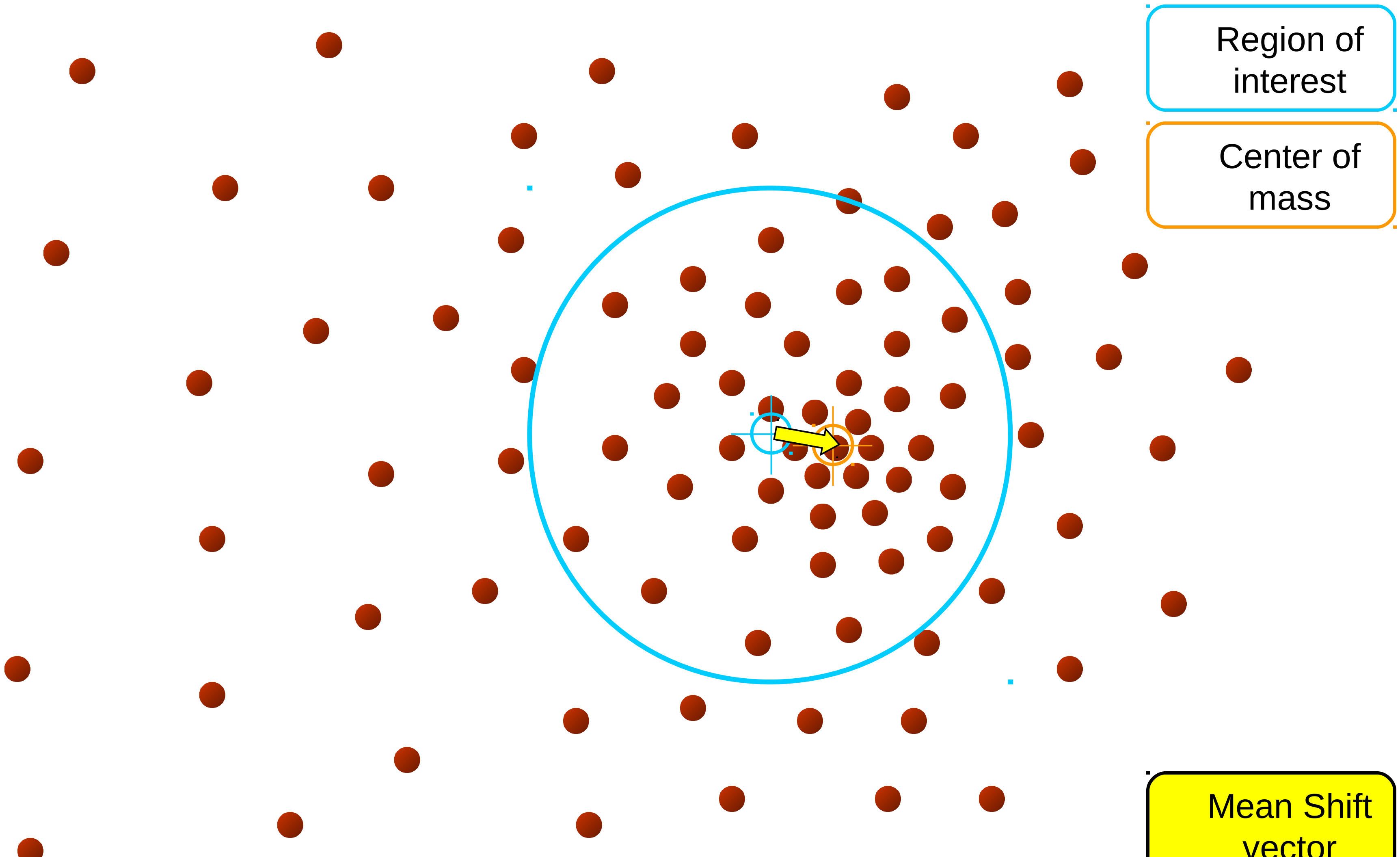


<u>Objective</u> : Find the densest region

Distribution of identical billiard balls

Slide Credit: **Y. Ukrainitz & B. Sarel**

Intuitive Description

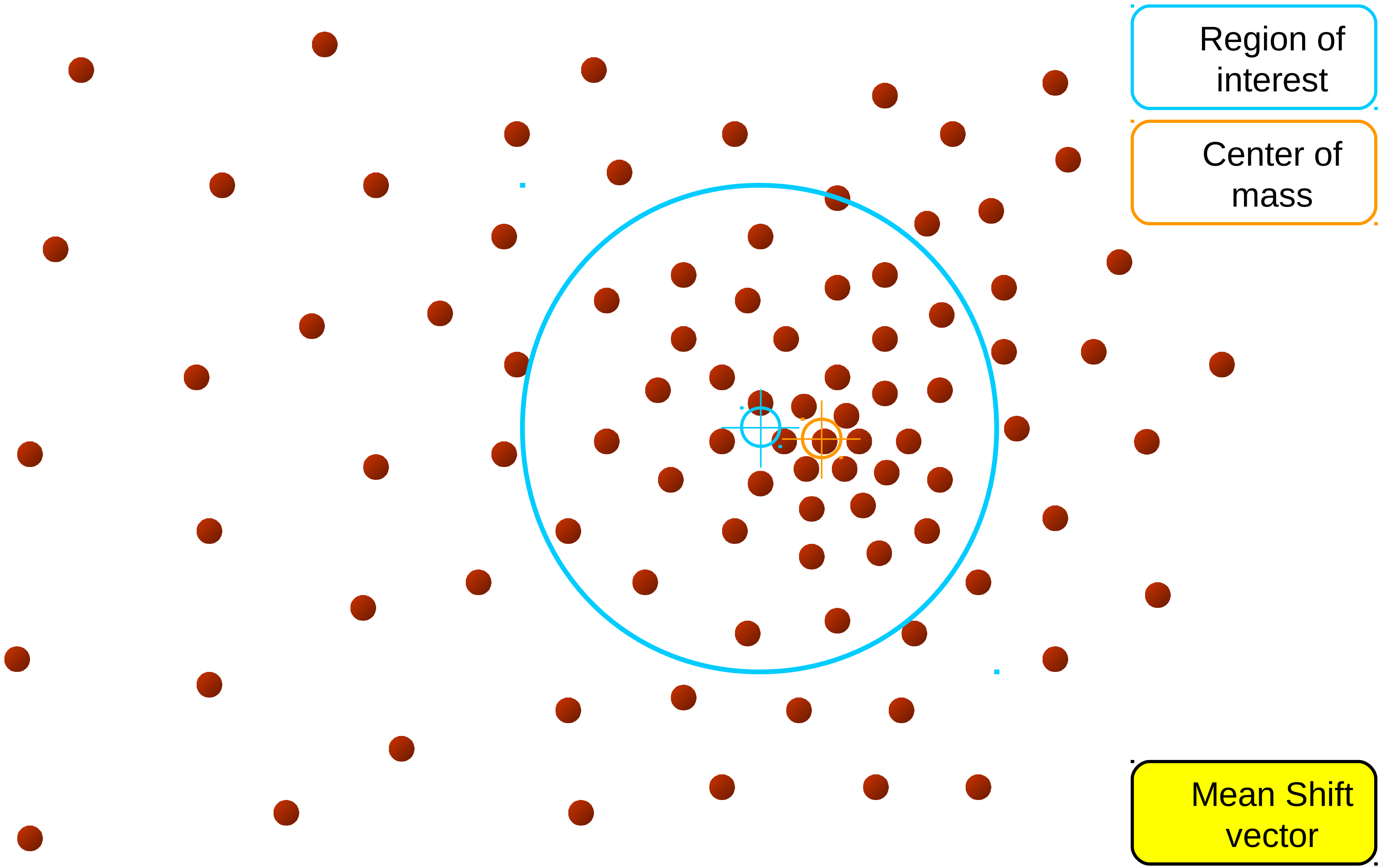


Objective : Find the densest region

Distribution of identical billiard balls

Slide Credit: **Y. Ukrainitz & B. Sarel**

Intuitive Description

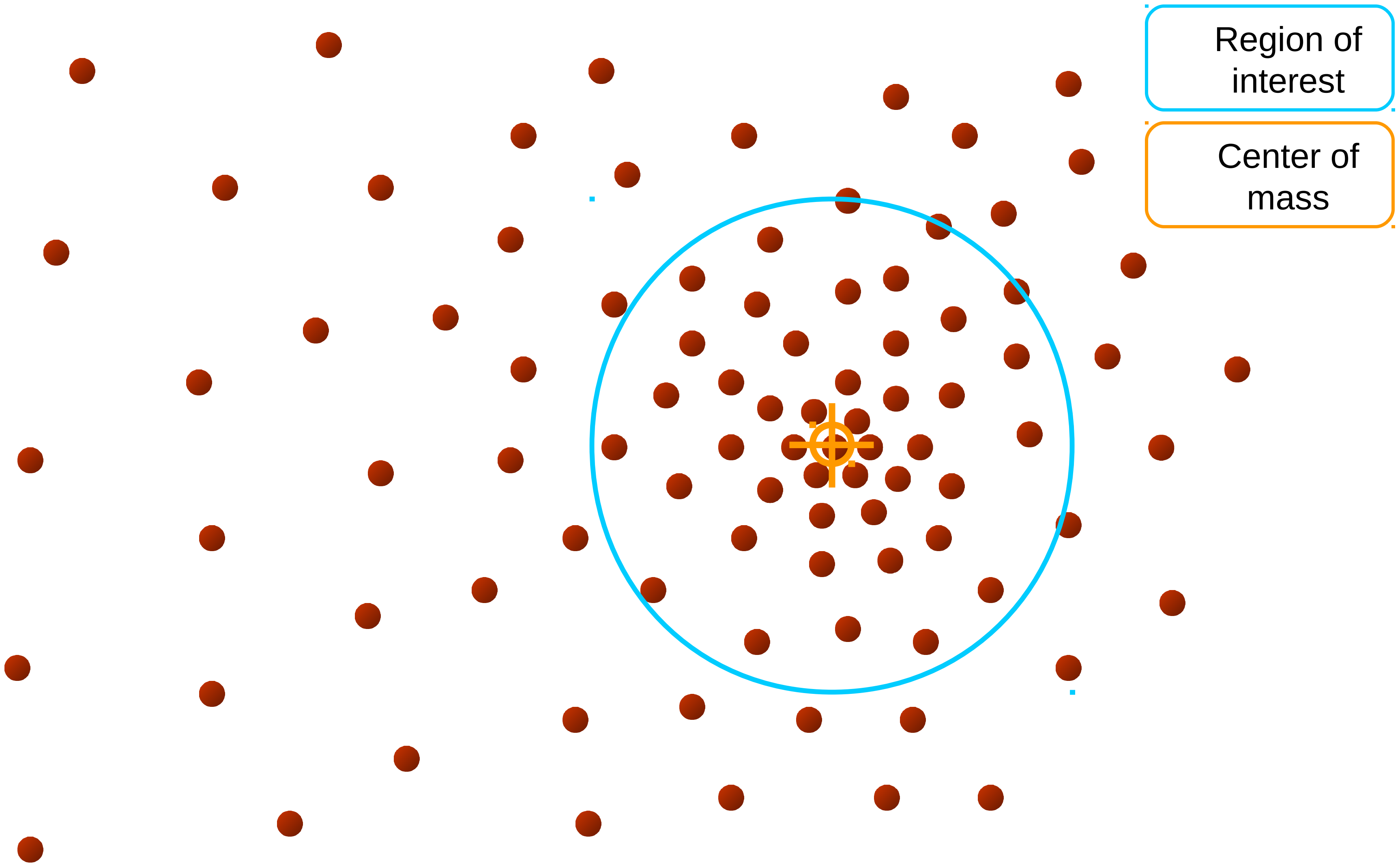


<u>Objective</u> : Find the densest region

Distribution of identical billiard balls

Slide Credit: **Y. Ukrainitz & B. Sarel**

Intuitive Description



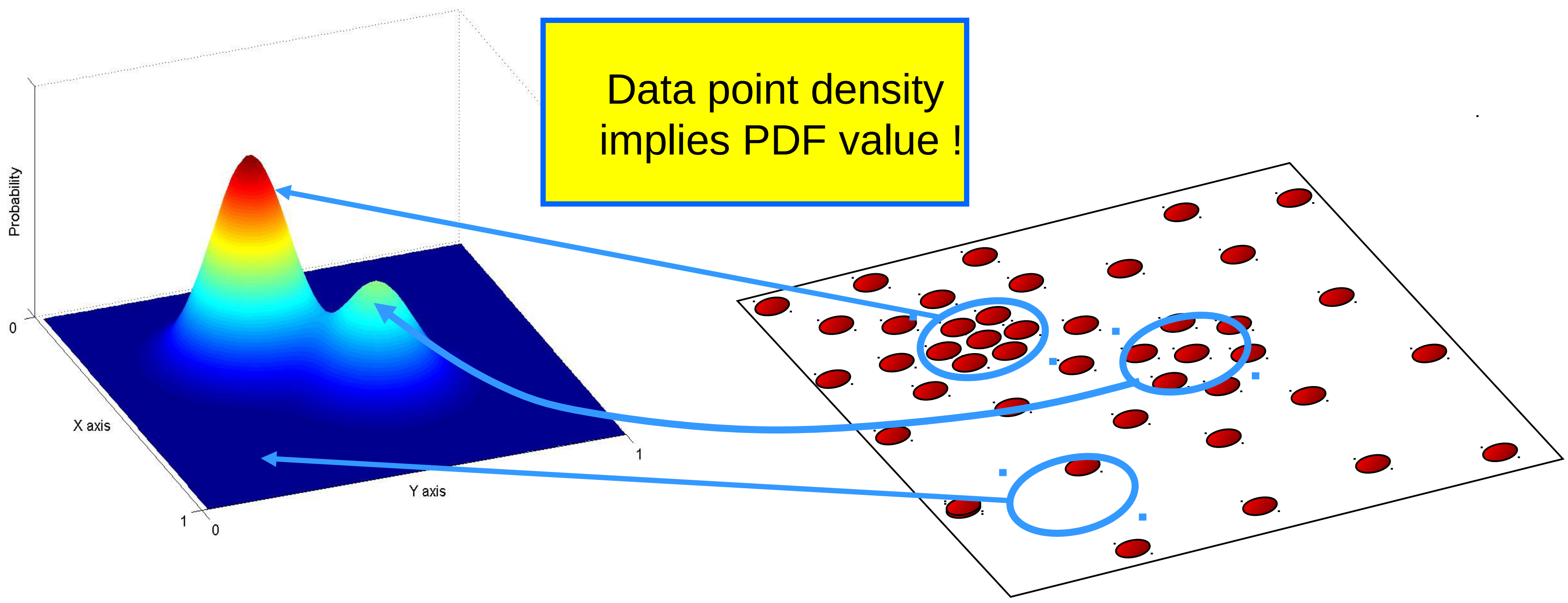
Objective : Find the densest region

Distribution of identical billiard balls

Slide Credit: **Y. Ukrainitz & B. Sarel**

Non-Parametric Density Estimation

<u>Assumption</u> : The data points are sampled from an underlying PDF



Assumed Underlying PDF

Real Data Samples

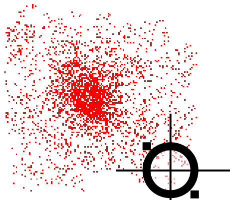
Slide Credit: Y. Ukrainitz & B. Sarel

Kernel Density Estimation

Various Kernels

$$P(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n} K(\mathbf{x} - \mathbf{x}_i)$$

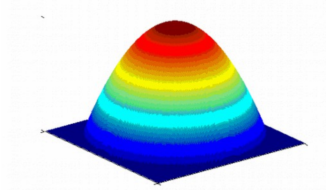
A function of some finite number of data points $x_1 \ldots x_n$



Examples:

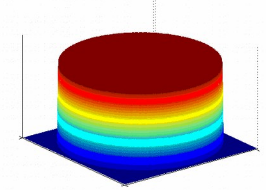
- Epanechnikov Kernel

$$K_E(\mathbf{x}) = \begin{cases} c\left(1 - \|\mathbf{x}\|^2\right) & \|\mathbf{x}\| \le 1 \\ 0 & \text{otherwise} \end{cases}$$



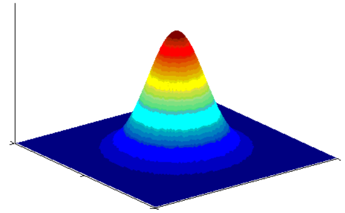
- Uniform Kernel

$$K_U(\mathbf{x}) = \begin{cases} c & \|\mathbf{x}\| \le 1 \\ 0 & \text{otherwise} \end{cases}$$



- Normal Kernel

$$K_N(\mathbf{x}) = c \cdot \exp\left(-\frac{1}{2}\|\mathbf{x}\|^2\right)$$



Slide Credit: Y. Ukrainitz & B. Sarel

Using Mean-Shift on Color Models

Two approaches:

1) Create a color "likelihood" image, with pixels weighted by similarity to the desired color (best for unicolored objects)

2) Represent color distribution with a histogram. Use mean-shift to find region that has most similar distribution of colors.

 Slide Credit: Y. Ukrainitz & B. Sarel

Robert Collins
CSE598C, PSU

Mean-shift on Weight Images

Ideally, we want an indicator function that returns 1 for pixels on the object we are tracking, and 0 for all other pixels

Instead, we compute likelihood maps where the value at a pixel is proportional to the likelihood that the pixel comes from the object we are tracking.

Computation of likelihood can be based on
- color
- texture
- shape (boundary)
- predicted location

 Slide Credit:

Example: Face Tracking using Mean -Shift

Gray Bradski, "Computer Vision Face Tracking for use in a Perceptual User Interface," ***IEEE Workshop On Applications of Computer Vision*****, Princeton, NJ, 1998, pp.214-219.**

Figure 7: Orientation of the flesh probability distribution marked on the source video image

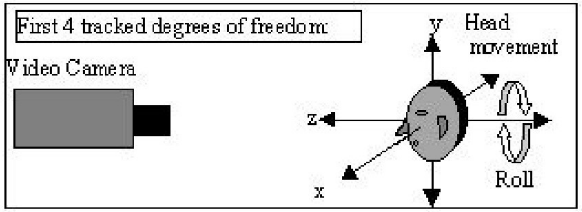
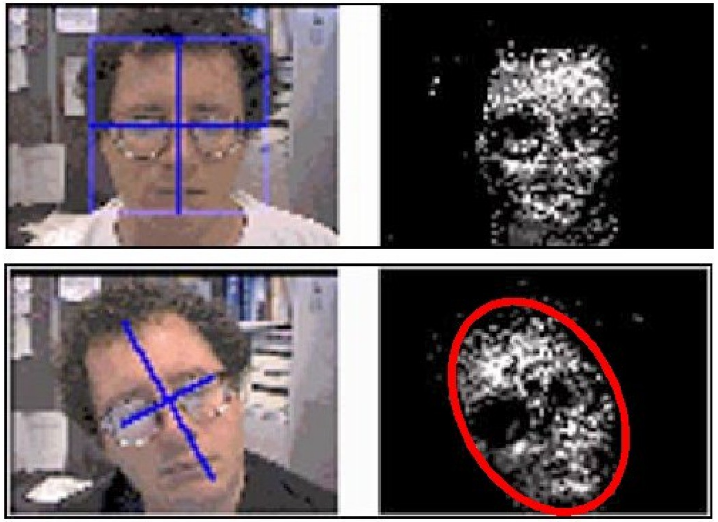


Figure 8: First four head tracked degrees of freedom: X, Y, Z location, and head roll

Slide Credit:

Robert Collins
CSE598C, PSU

Bradski's CamShift



X,Y location of mode found by mean-shift.
Z, Roll angle determined by fitting an ellipse to the mode found by mean-shift algorithm.

Slide Credit:

Robert Collins
CSE598C, PSU

Using Mean-Shift on Color Models

Two approaches:

1) Create a color "likelihood" image, with pixels weighted by similarity to the desired color (best for unicolored objects)

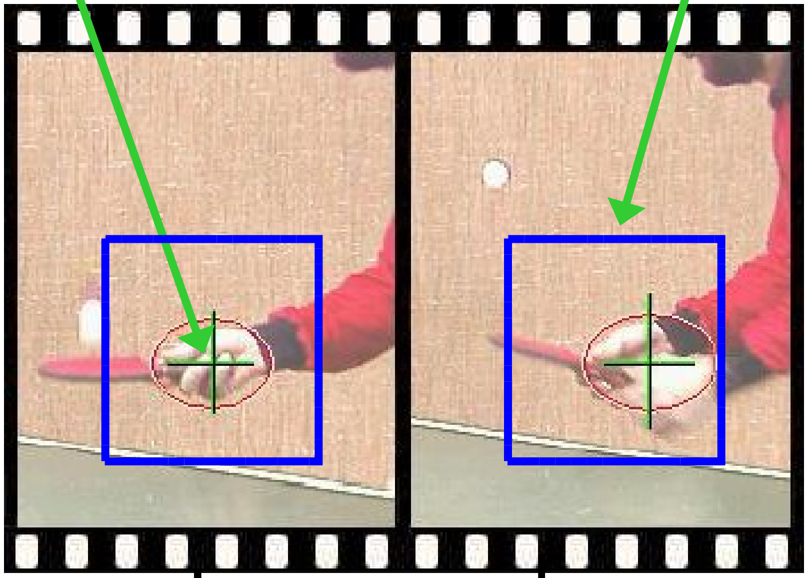
2) Represent color distribution with a histogram. Use mean-shift to find region that has most similar distribution of colors.

 Slide Credit:

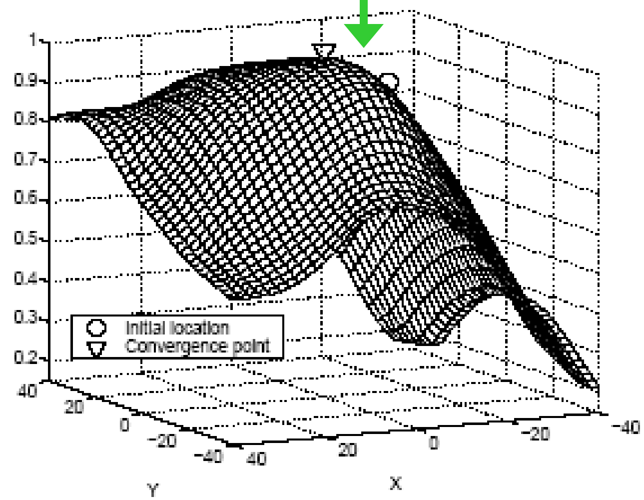
Mean-Shift Object Tracking

Target Localization Algorithm

Start from the position of the model in the current frame → Search in the model's neighborhood in next frame → Find best candidate by maximizing a similarity func.



$\vec{q}$ $\vec{p}(y)$

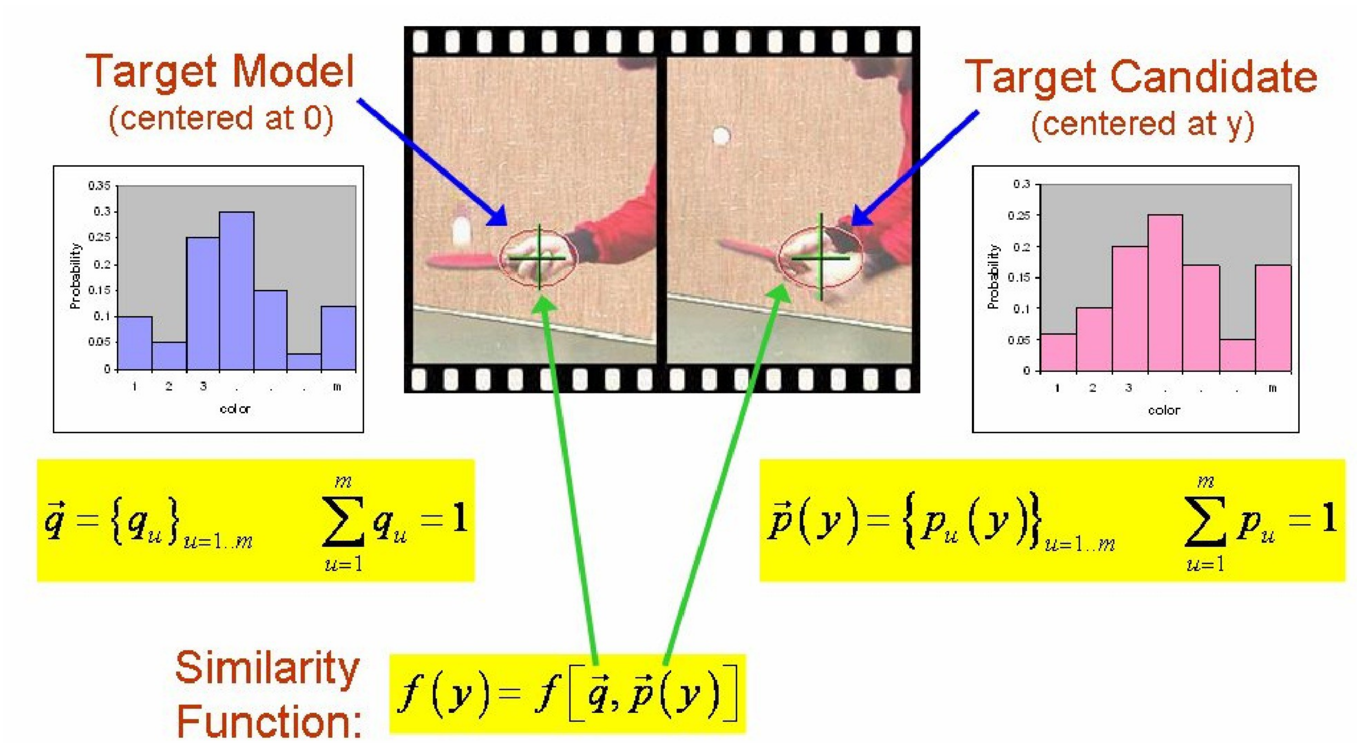


$f\left[\vec{p}(y),\vec{q}\right]$

Slide Credit: **Y. Ukrainitz & B. Sarel**

Mean-Shift Object Tracking

PDF Representation



Ukrainitz&Sarel, Weizmann

Slide Credit: **Y. Ukrainitz & B. Sarel**

Robert Collins
CSE598C, PSU

Glossing over the Details

Spatial smoothing of similarity function by introducing a spatial kernel (Gaussian, box filter)

Take derivative of similarity with respect to colors. This tells what colors we need more/less of to make current hist more similar to reference hist.

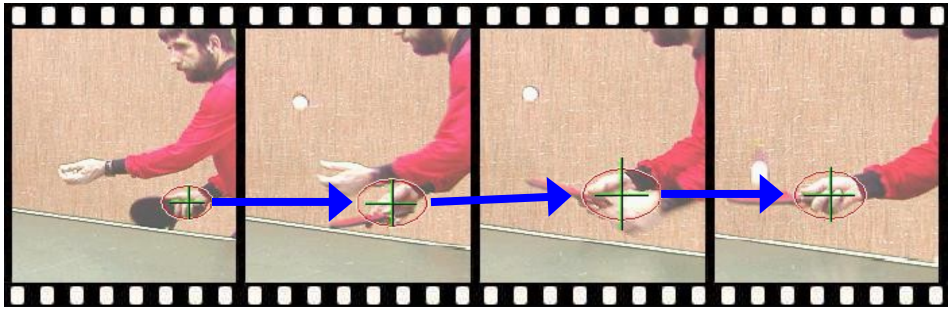
Result is weighted mean shift we used before. However, the color weights are now computed "on-the-fly", and change from one iteration to the next.

Mean-Shift Object Tracking

Adaptive Scale

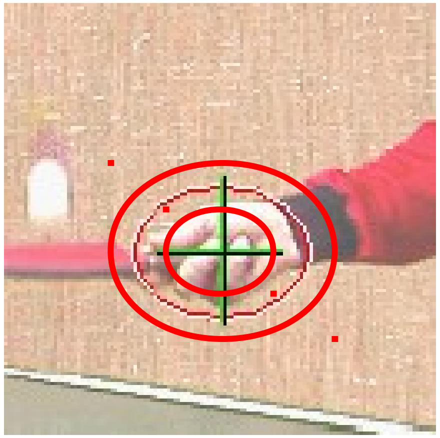
Problem:

The scale of the target changes in time → The scale (*h*) of the kernel must be adapted



Solution:

Run localization 3 times with different *h* → Choose *h* that achieves maximum similarity



Slide Credit: Y. Ukrainitz & B. Sarel

Mean-Shift Object Tracking

Results

Feature space: 16×16×16 quantized RGB
Target: manually selected on 1st frame
Average mean-shift iterations: 4

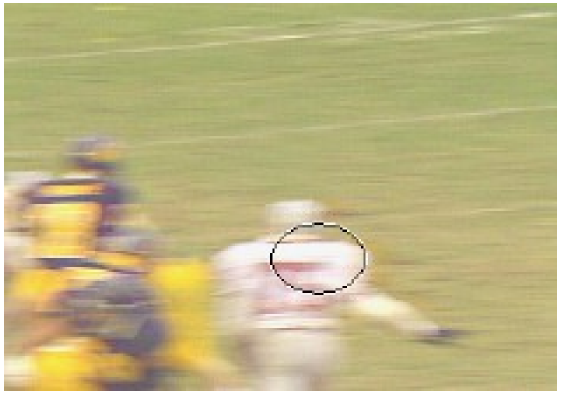
 Slide Credit:

Mean-Shift Object Tracking

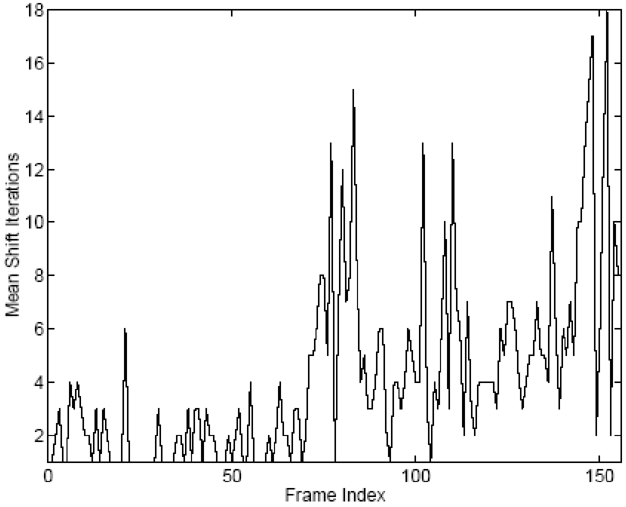
Results

Partial occlusion

Distraction



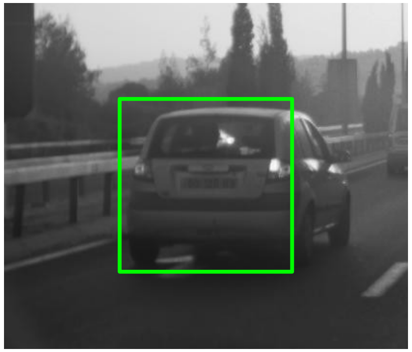
Motion blur



Slide Credit: **Y. Ukrainitz & B. Sarel**

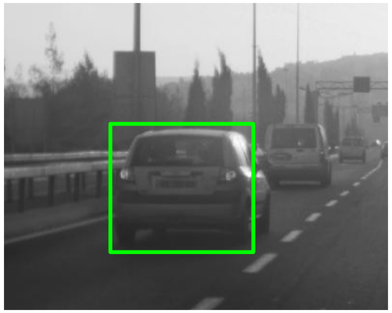
Tracking a Sequence

Original Template
(t_0)



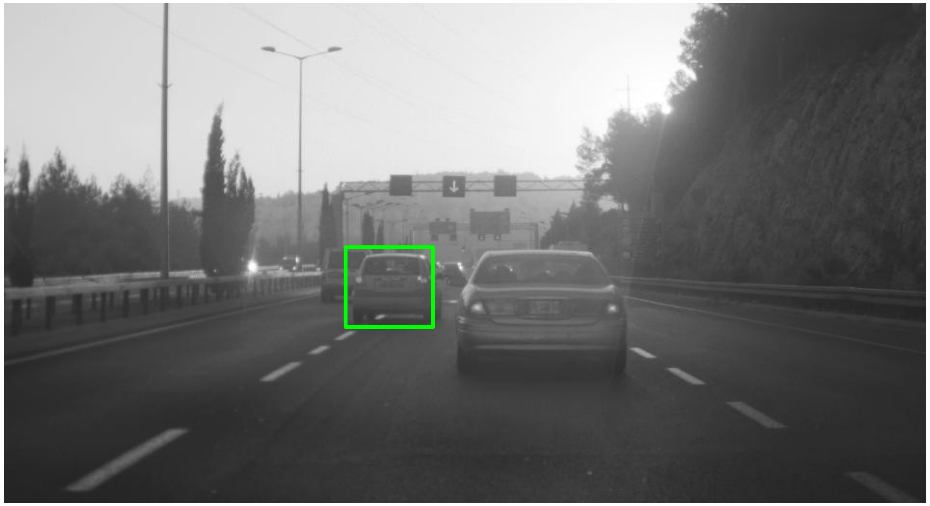
. . .

Later Frame
(t_10)



. . .

Current Frame
(t_N)



Which previous frame to use as Template for Current frame?
- update allows handling changes in appearance
- update may produce drift

Slide Credit:

Visual Tracking with Online Multiple Instance Learning

Boris Babenko[1], Ming-Hsuan Yang[2], Serge Belongie[1]

1. University of California, San Diego
2. University of California, Merced

Tracking by Detection

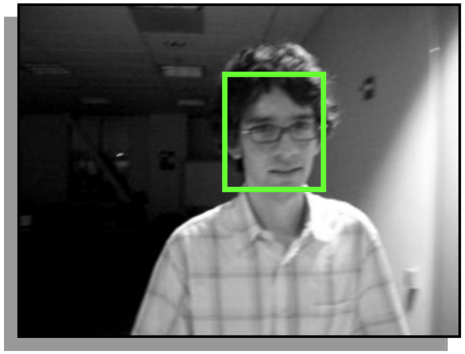
- Recent tracking work
  - Focus on appearance model
  - Borrow techniques from obj. detection
    - Slide a discriminative classifier around image
  - **Adaptive** appearance model

[Collins et al. '05, Grabner et al. '06, Ross et al. '08]

Slide Credit: **Babenko, Yang, Belongie**

Tracking by Detection

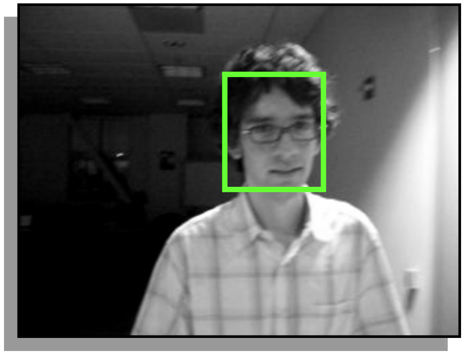
- First frame is labeled



 Slide Credit: **Babenko, Yang, Belongie**

Tracking by Detection

- First frame is labeled

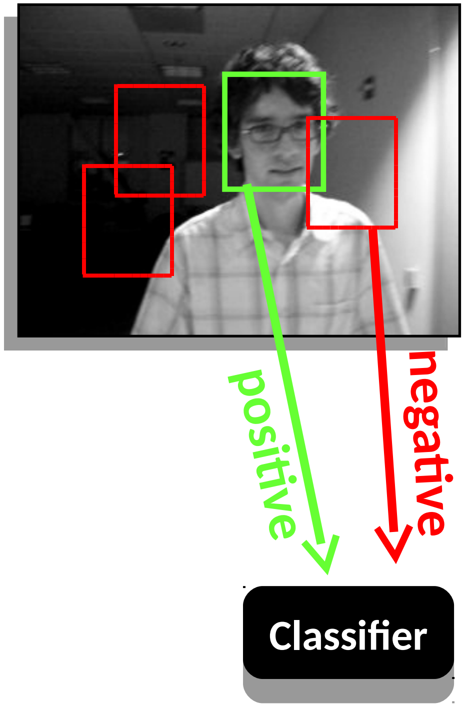


Online classifier (i.e. Online AdaBoost)

Slide Credit: **Babenko, Yang, Belongie**

Tracking by Detection

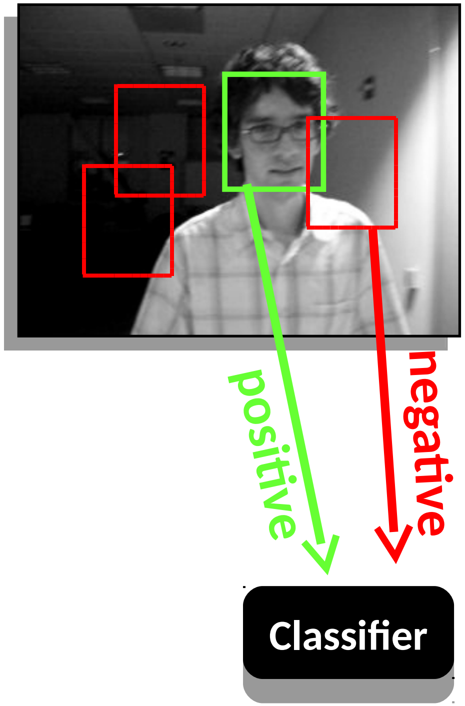
- Grab one **positive** patch, and some **negative** patch, and train/update the model.



Slide Credit: **Babenko, Yang, Belongie**

Tracking by Detection

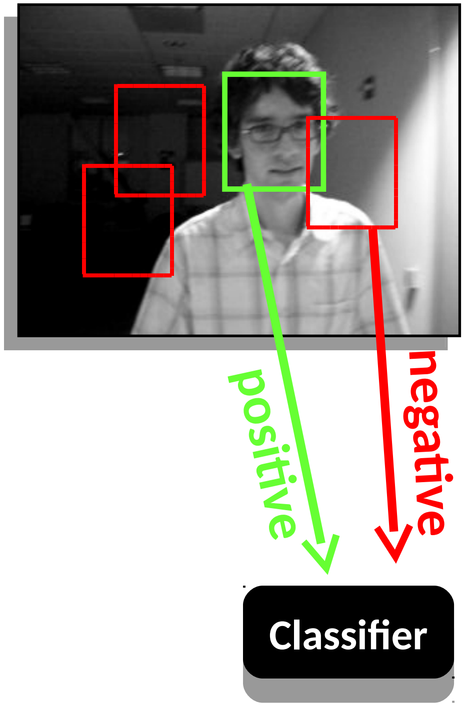
- Get next frame

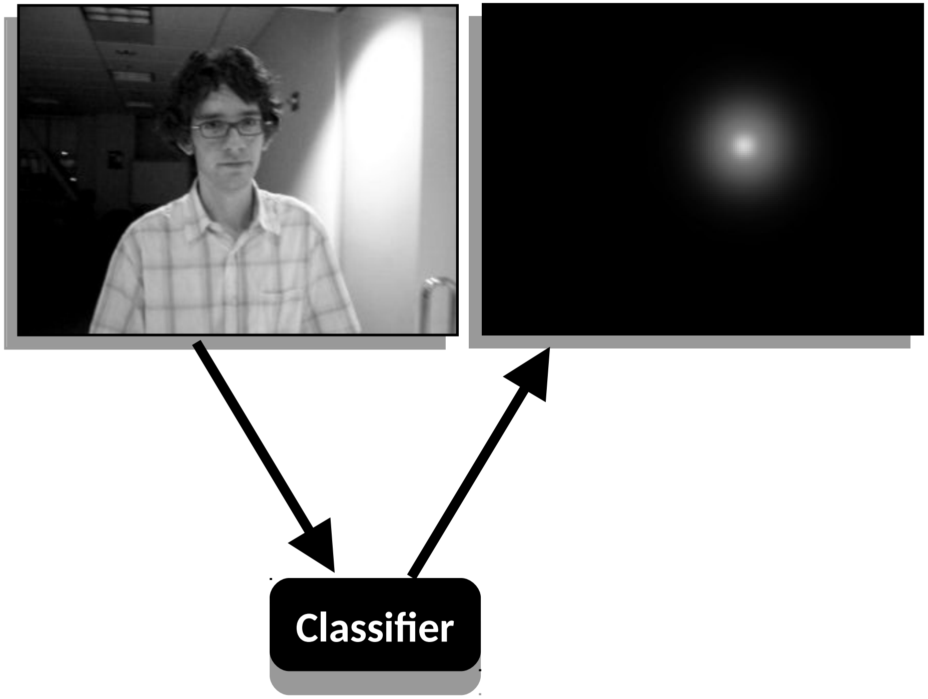


 Slide Credit: **Babenko, Yang, Belongie**

Tracking by Detection

- Evaluate classifier in some search window

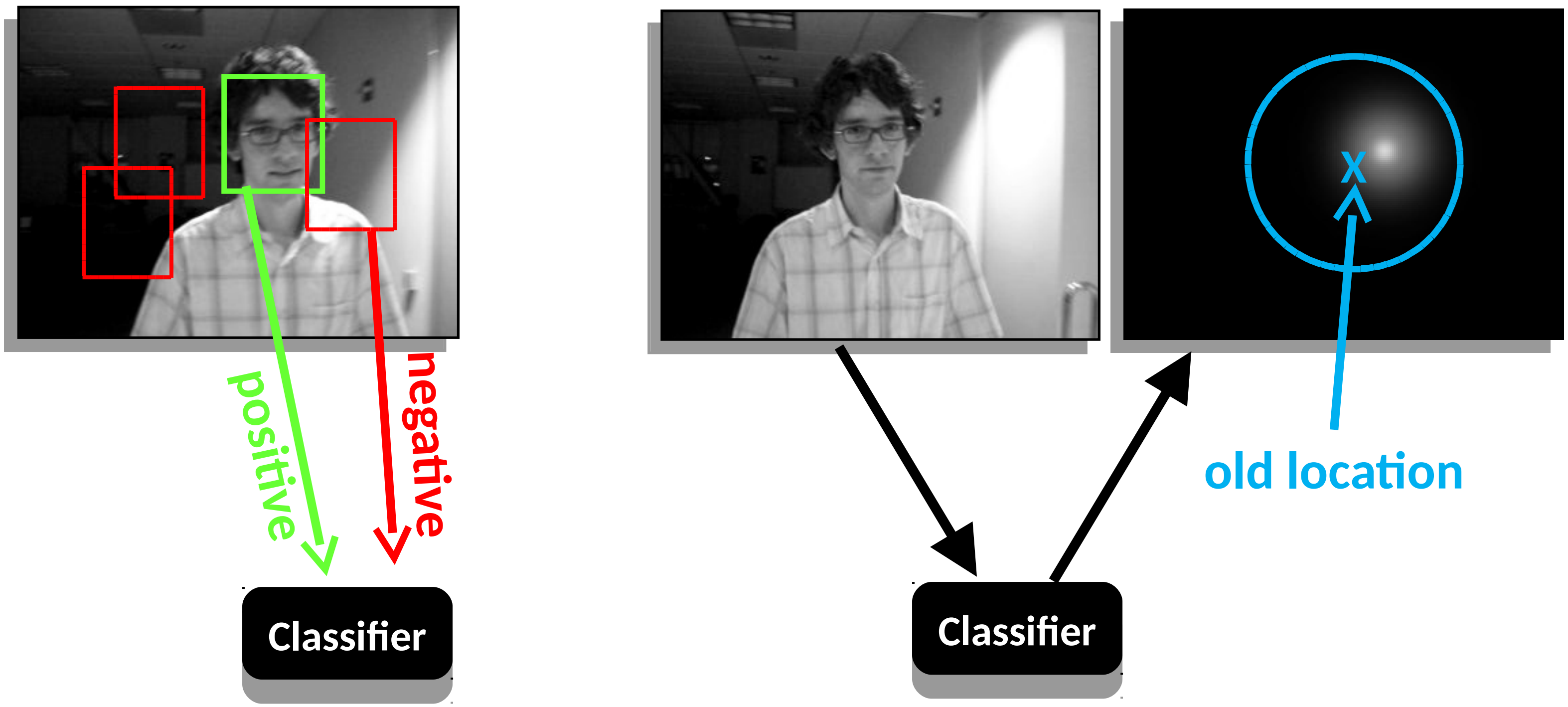




 Slide Credit: Babenko, Yang, Belongie

Tracking by Detection

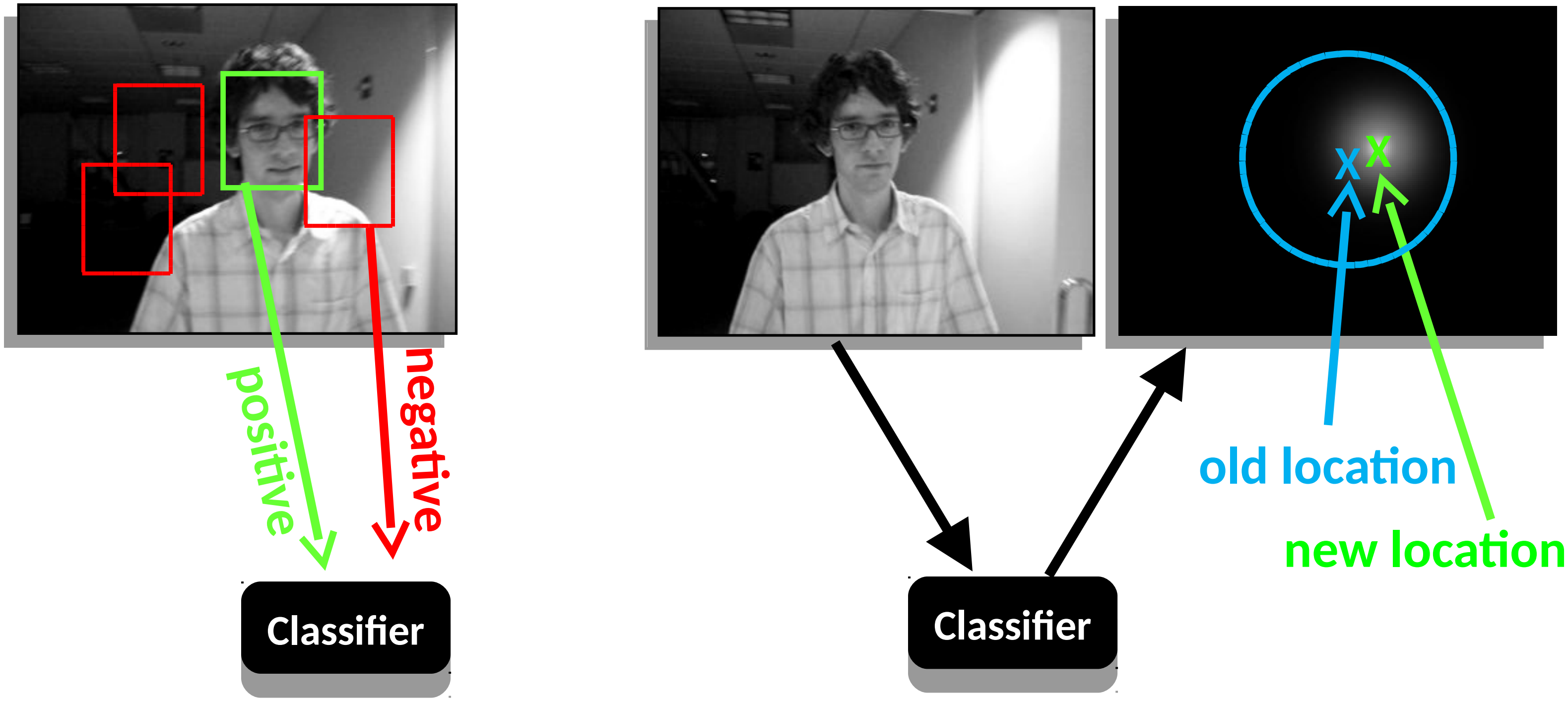
- Evaluate classifier in some search window



 Slide Credit: **Babenko, Yang, Belongie**

Tracking by Detection

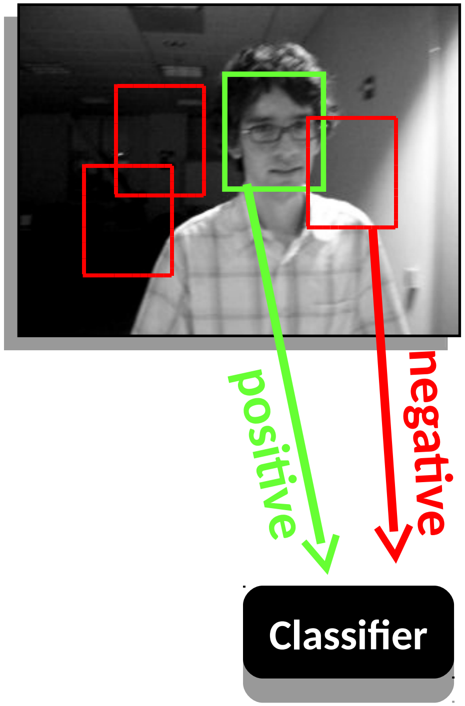
- Find max response

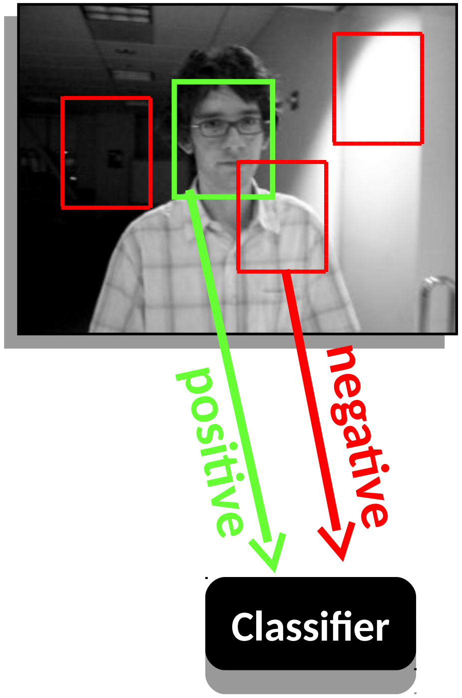


 Slide Credit: **Babenko, Yang, Belongie**

Tracking by Detection

- Repeat...

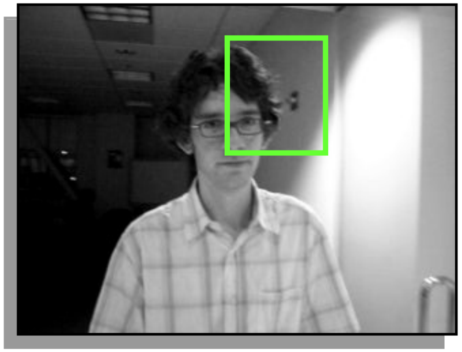




 Slide Credit: **Babenko, Yang, Belongie**

Problems with Adaptive Appearance Models

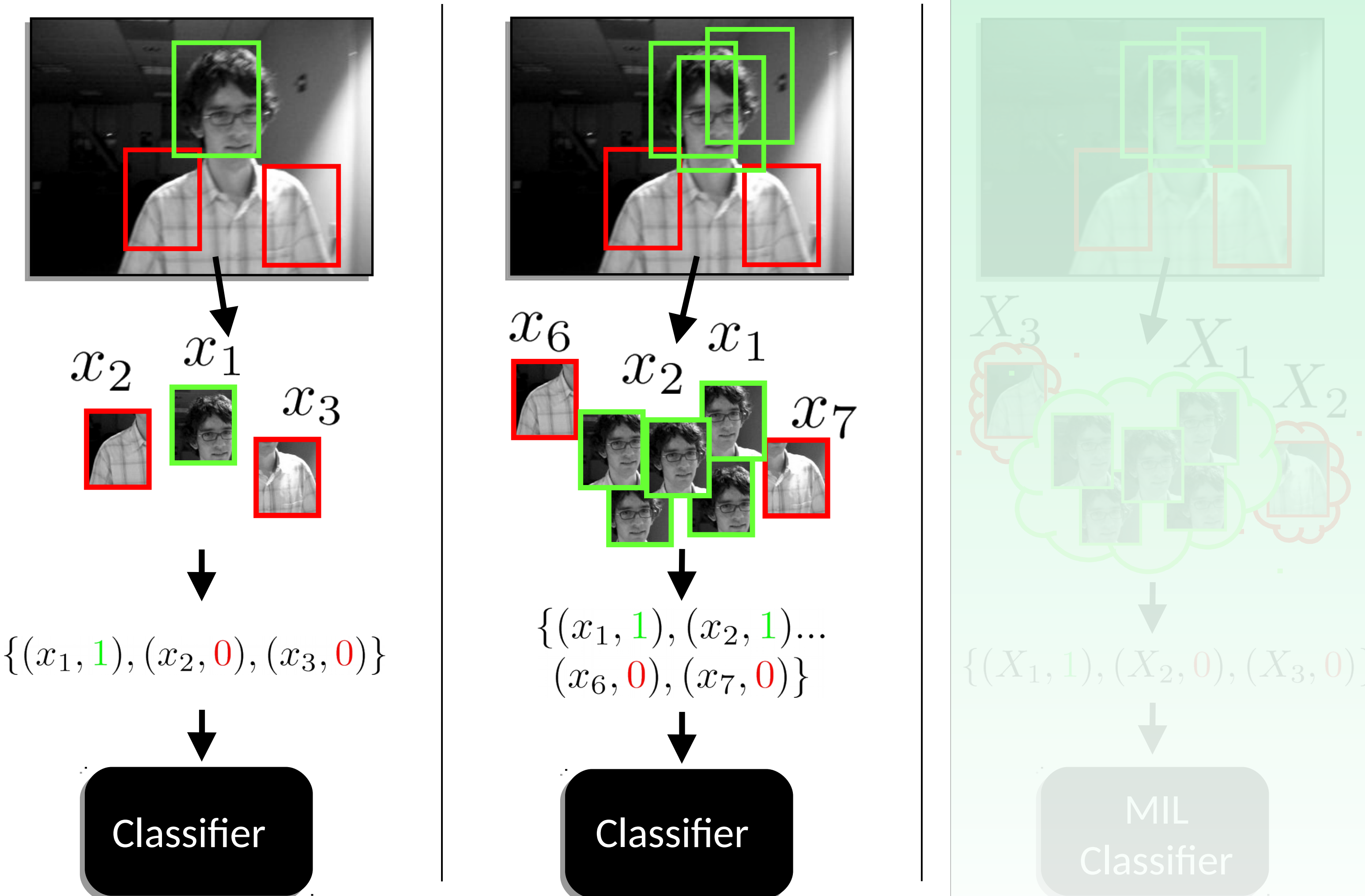
- What if classifier is a bit off?
  - Tracker starts to drift



- How to choose training examples?

 Slide Credit: **Babenko, Yang, Belongie**

How to Get Training Examples



Slide Credit: Babenko, Yang, Belongie

Multiple Instance Learning (MIL)

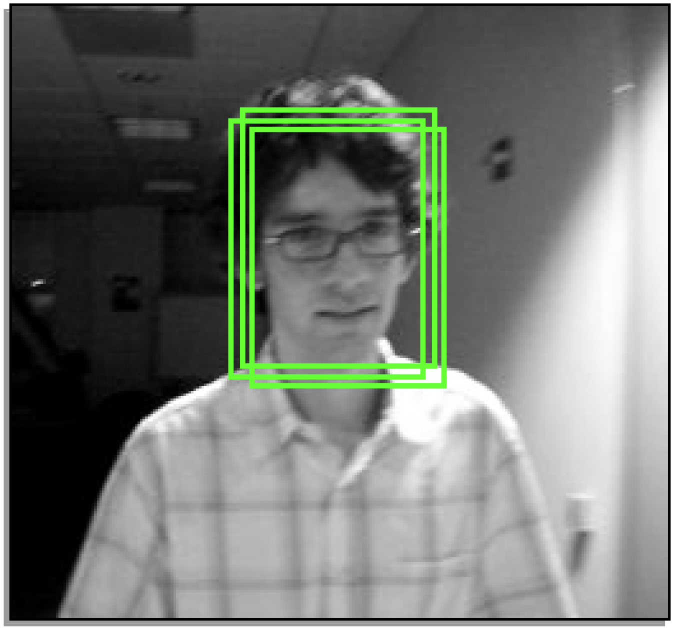
- Ambiguity in training data
- Instead of instance/label pairs, get **bag of instances**/label pairs
- Bag is **positive** if one or more of it's members is positive

[Keeler '90, Dietterich et al. '97]

Slide Credit: **Babenko, Yang, Belongie**

Object Detection

- Problem:
  - Labeling with rectangles is inherently ambiguous
  - Labeling is sloppy

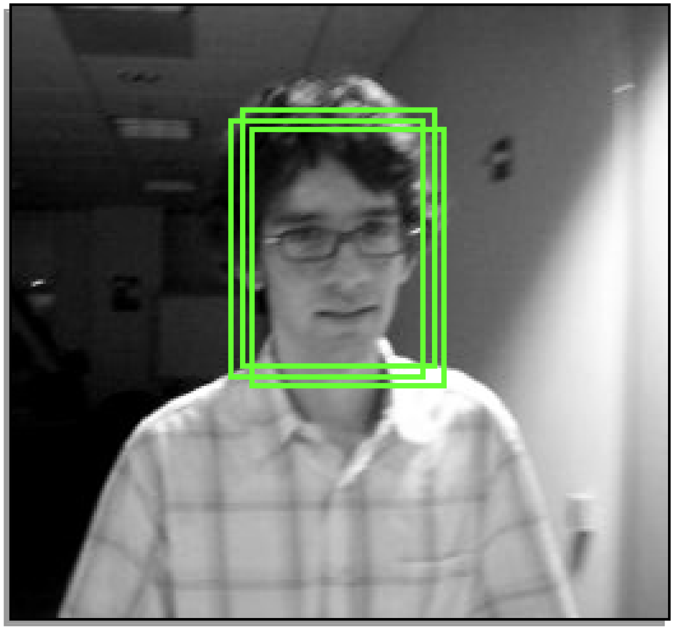


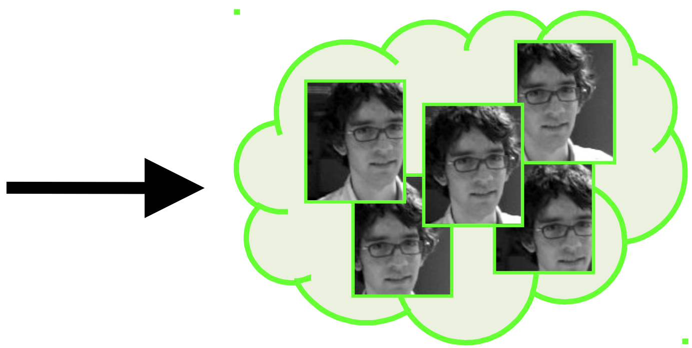
[Viola et al. '05]

Slide Credit: **Babenko, Yang, Belongie**

MIL for Object Detection

- Solution:
  - Take all of these patches, put into **positive** **bag**
  - At least one patch in bag is “correct”





[Viola et al. ‘05]

Slide Credit: **Babenko, Yang, Belongie**

Multiple Instance Learning (MIL)

- Supervised Learning Training Input

$$\{x_1, \ldots, x_n\}, x_i \in \mathcal{X}$$

$$\{y_1, \ldots, y_n\}, y_i \in \mathcal{Y}$$

- MIL Training Input

$$\{X_1, \ldots, X_n\}, X_i = \{x_{i1}, \ldots, x_{im}\}$$

$$\{y_1, \ldots, y_n\}, y_i \in \{0, 1\}$$

Slide Credit: **Babenko, Yang, Belongie**

Multiple Instance Learning (MIL)

- **Positive** bag contains **at least one** positive instance

$$y_i = \max_j y_{ij}$$

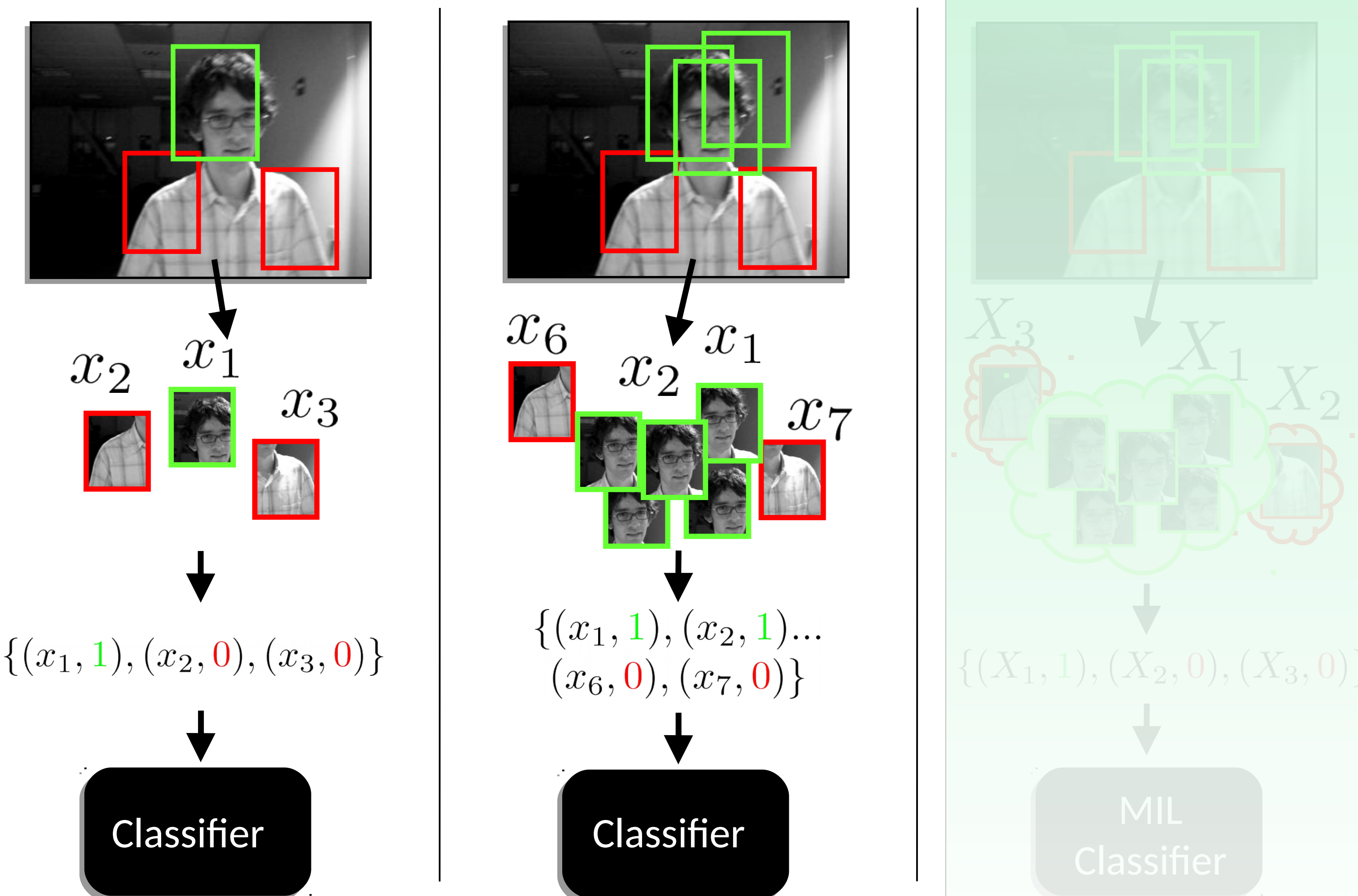
- Goal: learning **instance classifier**

$$h : \mathcal{X} \rightarrow \{0, 1\}$$

- Classifier is same format as standard learning

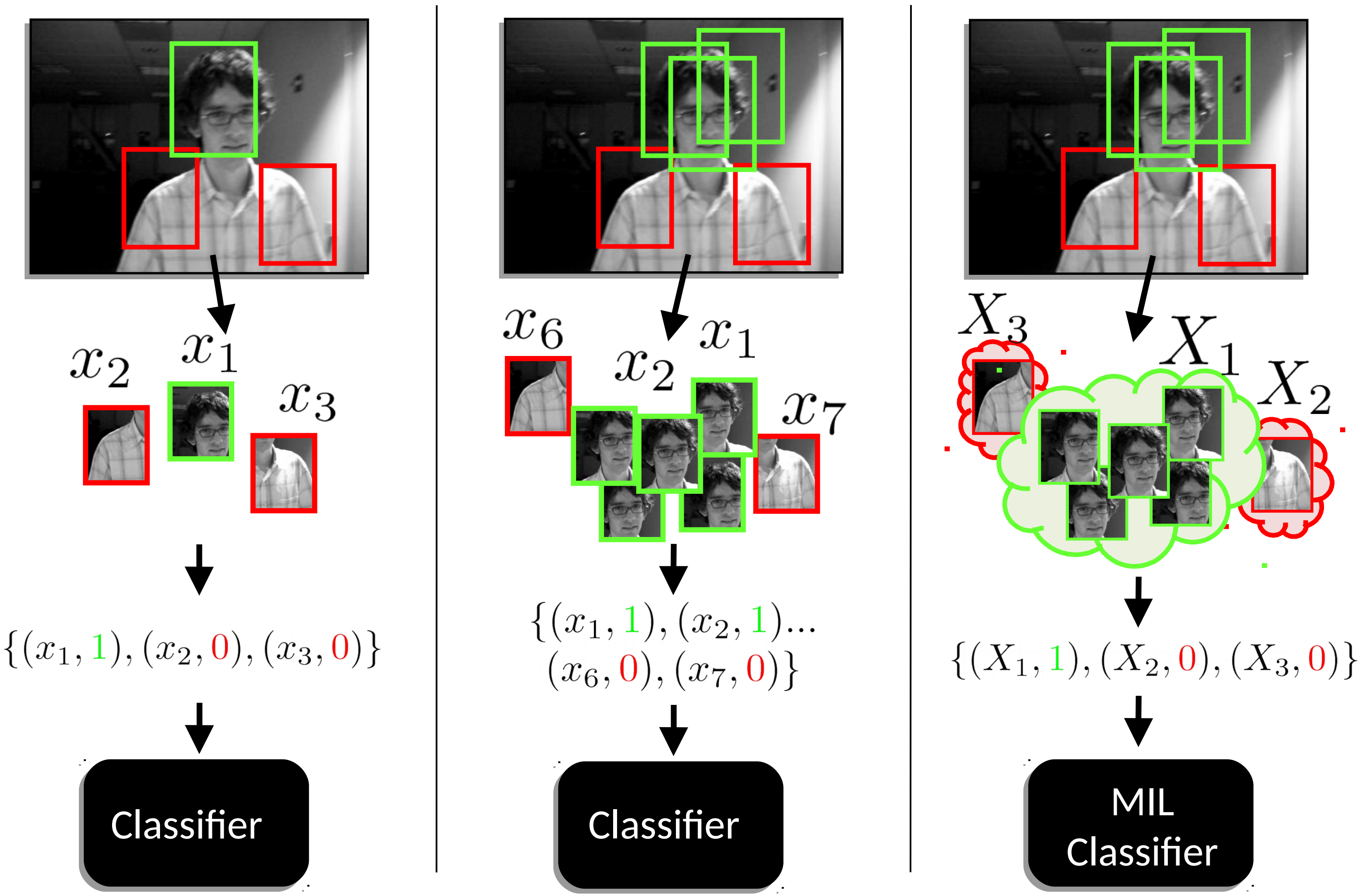
 Slide Credit: **Babenko, Yang, Belongie**

How to Get Training Examples



Slide Credit: **Babenko, Yang, Belongie**

How to Get Training Examples



Slide Credit: **Babenko, Yang, Belongie**

Online MILBoost

- Need an **online** MIL algorithm
- Combine ideas from **MILBoost** and **Online Boosting**

[Oza et al. '01, Viola et al. '05, Grabner et al. '06]

Slide Credit: **Babenko, Yang, Belongie**

Boosting

- Train classifier of the form:

$$\mathbf{H}_K(x) = \sum_{k=1}^{K} \mathbf{h}_k(x)$$

where $\mathbf{h}_k$ is a weak classifier

- Can make binary predictions using $\mathrm{sign}(\mathbf{H}_K(x))$

[Freund et al. '97]

Slide Credit: **Babenko, Yang, Belongie**

MILBoost

- **Objective to maximize:** Log likelihood of bags:

$$\mathcal{L}(\mathbf{H}) = \sum_i y_i \log(p_i) + (1 - y_i) \log(1 - p_i)$$

where

$$p_{ij} = \mathbb{P}(y_{ij} = 1 | x_{ij}) \equiv \sigma\left(\mathbf{H}(x)\right)$$ **(as in LogitBoost)**

$$p_i = \mathbb{P}(y_i = 1 | X_i) \equiv 1 - \prod_j (1 - p_{ij})$$ **(Noisy-OR)**

[Viola et al. '05, Friedman et al. '00]

Slide Credit: **Babenko, Yang, Belongie**

MILBoost

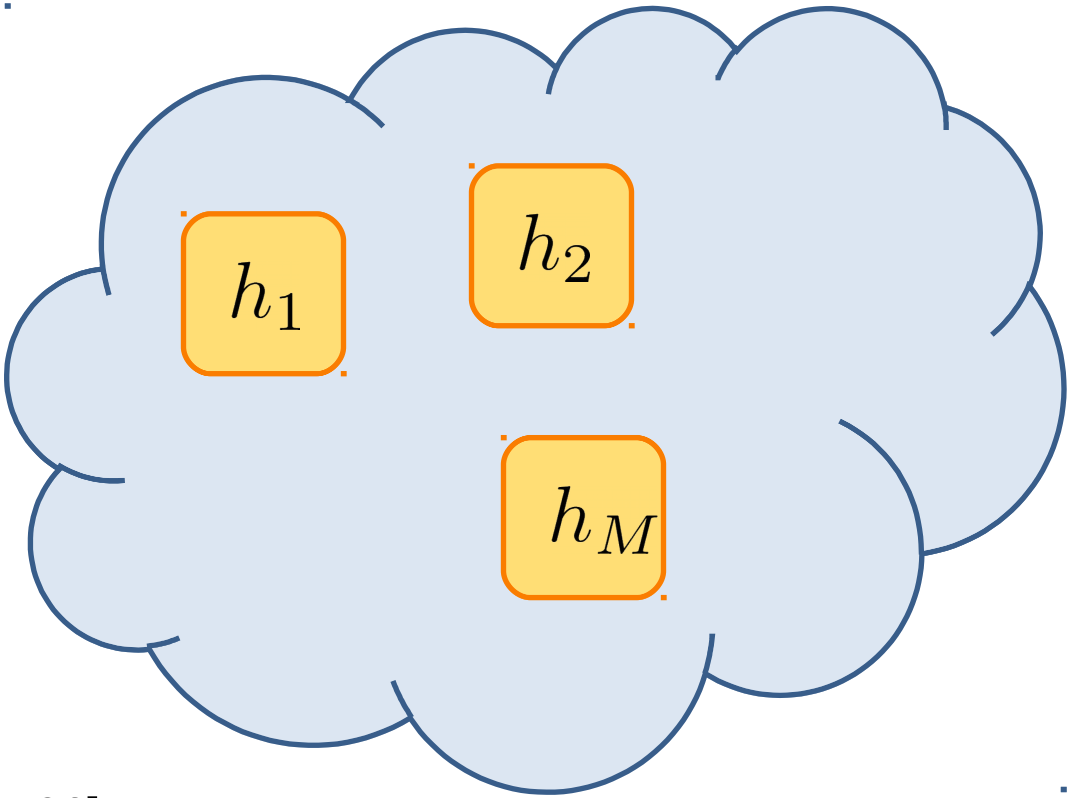
- Train weak classifier in a greedy fashion

$$\mathbf{h}_{k+1} = \underset{\mathbf{h}\in\mathcal{H}}{\operatorname{argmax}}\, \mathcal{L}(\mathbf{H}_k + \mathbf{h})$$

- For batch **MILBoost** can optimize using functional gradient descent.
- We need an **online** version…

Online MILBoost

- At all times, keep a pool of $M >> K$ weak classifier candidates

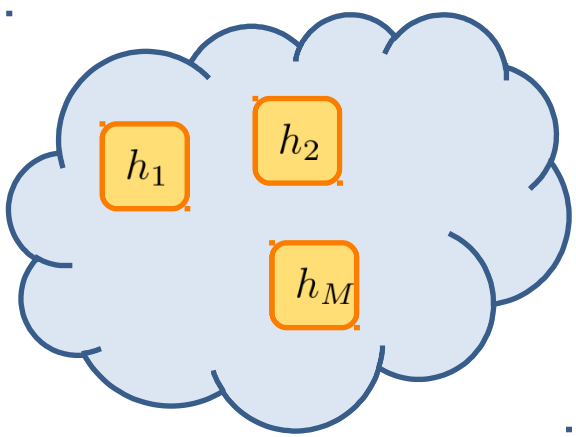


[Grabner et al. '06]

Slide Credit: **Babenko, Yang, Belongie**

Updating Online MILBoost

- At time t get more training data
  - Update all candidate classifiers
  - Pick best K in a greedy fashion



$$\mathbf{h}_{k+1} = \underset{\mathbf{h} \in \{h_1, h_2, \ldots h_M\}}{\operatorname{argmax}} \mathcal{L}(\mathbf{H}_k + \mathbf{h})$$

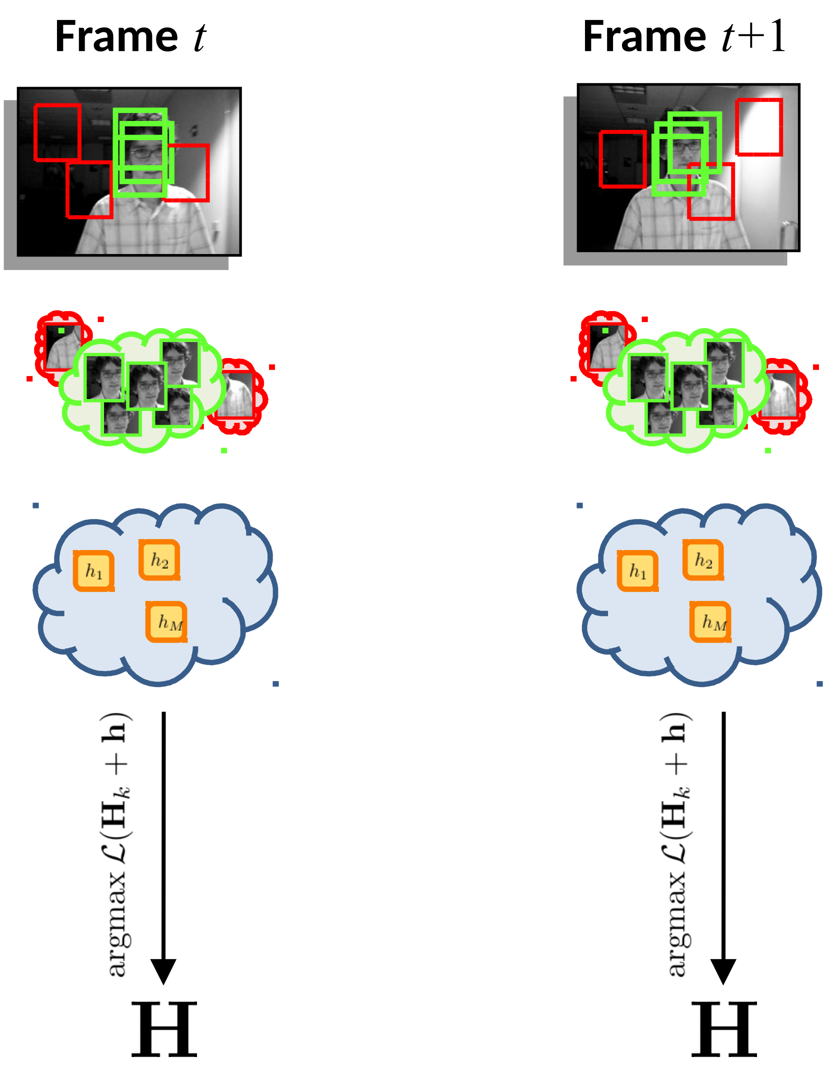
 Slide Credit: **Babenko, Yang, Belongie**

Online MILBoost

Get data (bags)

Update all classifiers in pool

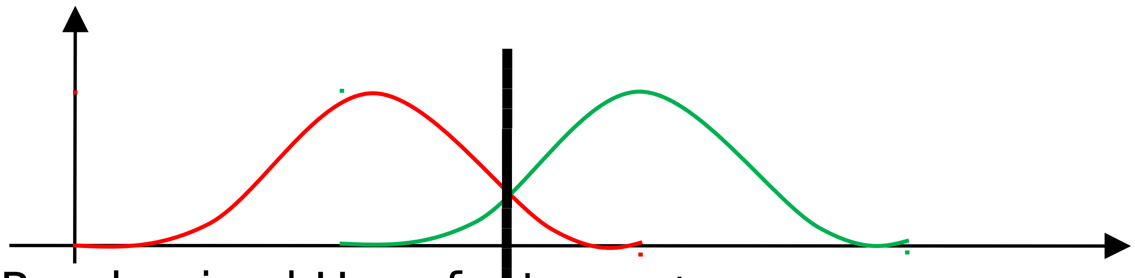
Greedily add best *K* to strong classifier



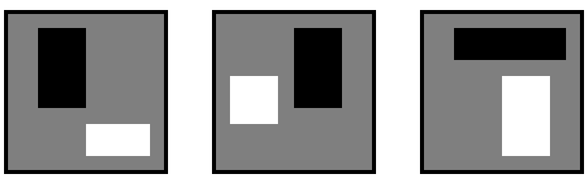
Slide Credit: Babenko, Yang, Belongie

MILTrack

- **MILTrack =**
  - Online MILBoost +
  - Stumps for weak classifiers +



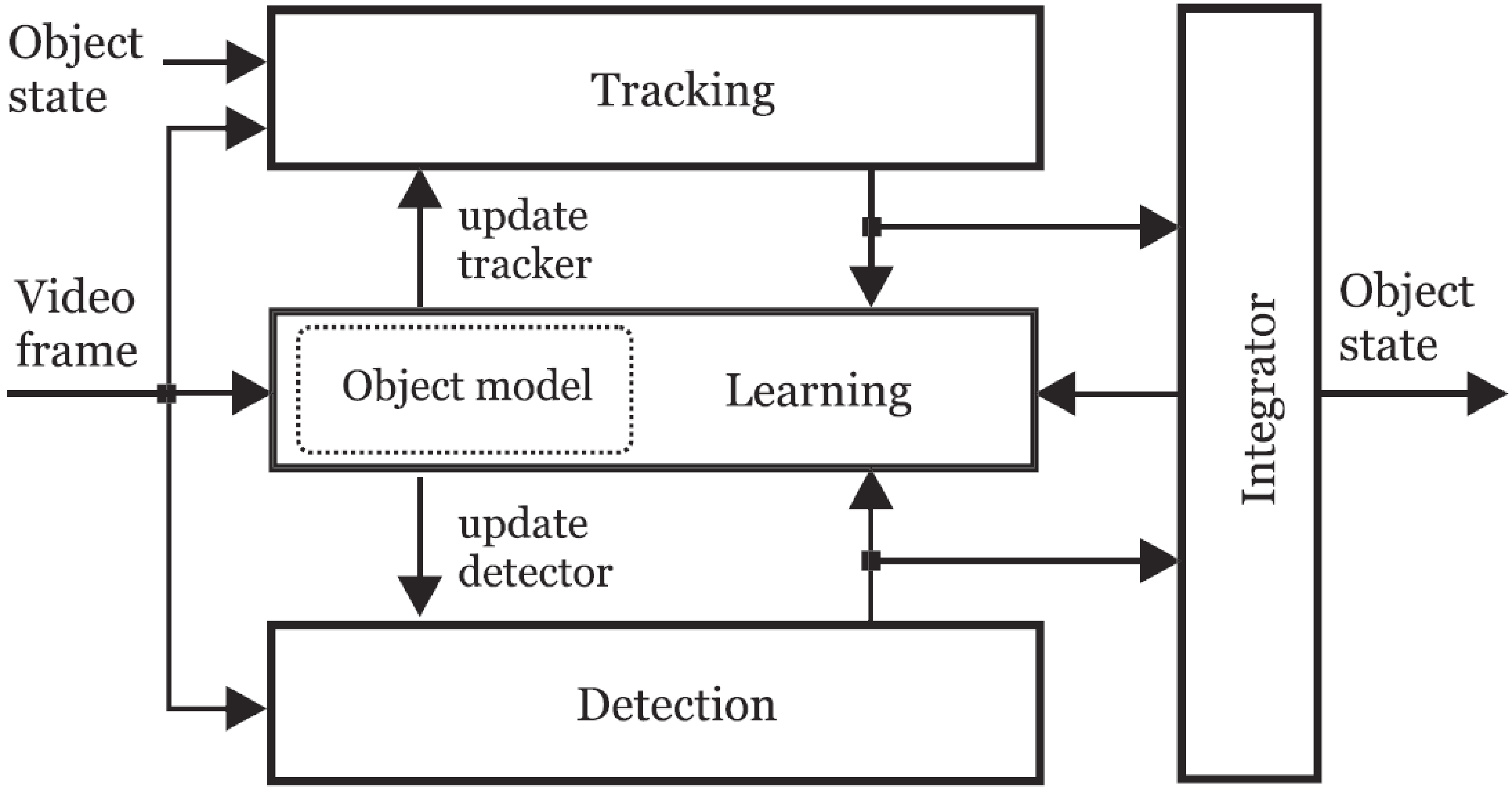
  - Randomized Haar features +



  - Simple motion model + greedy local search

 Slide Credit: **Babenko, Yang, Belongie**

Alternate Method: Tracking-Learning-Detection



[“Tracking-Learning-Detection” Zdenek Kalal, Krystian Mikolajczyk, and Jiri Matas]

 Slide Credit:

Alternate Method: Tracking-Learning-Detection

Learns 2 types of "experts":

- P-expert : identifies only false negatives
- N-expert : identifies only false positives

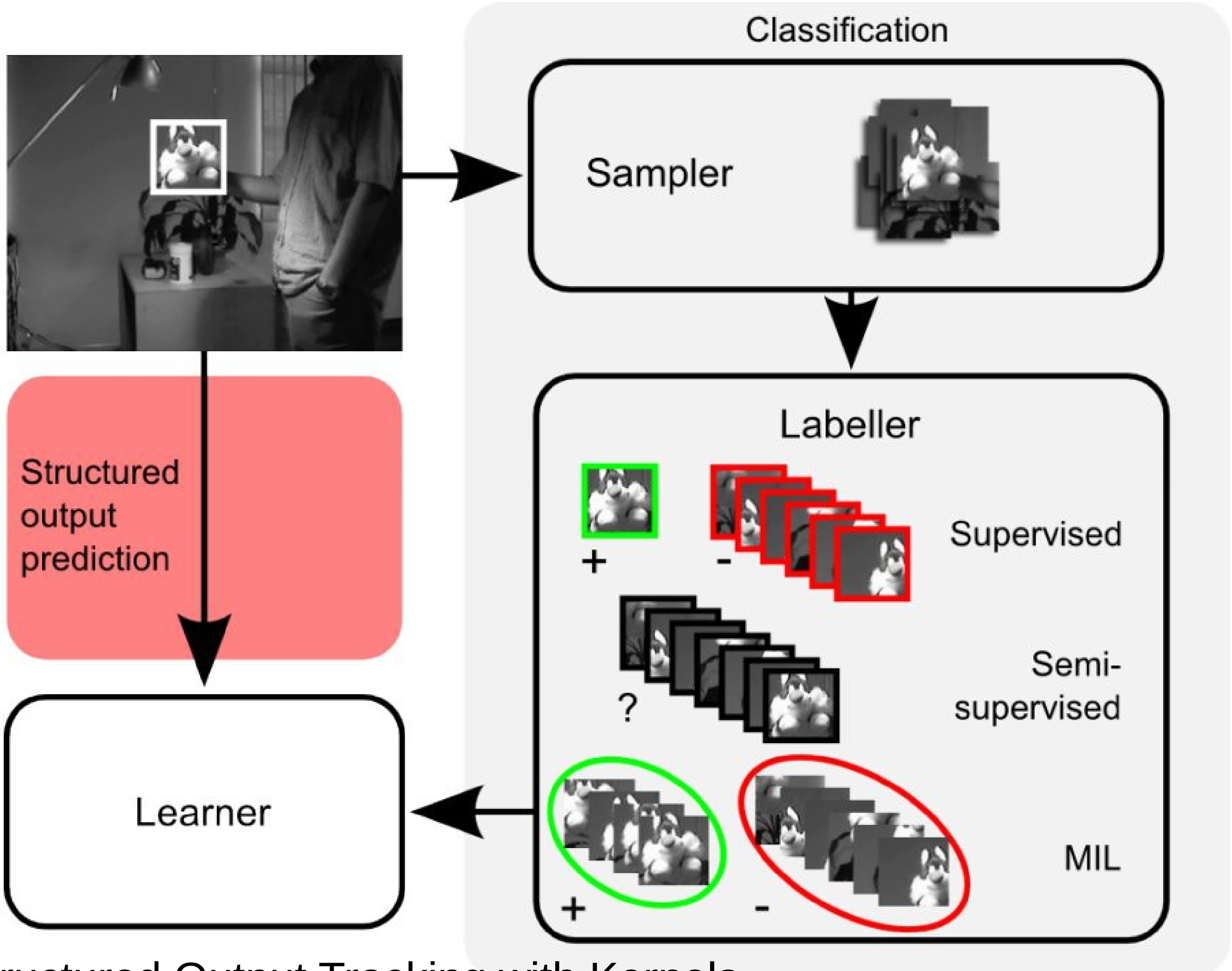
Both of the experts make errors themselves

However, their independence enables mutual compensation of their errors

["Tracking-Learning-Detection" Zdenek Kalal, Krystian Mikolajczyk, and Jiri Matas]

Slide Credit:

Alternate Method 2: Struck Tracker



Struck: Structured Output Tracking with Kernels
Sam Hare, Amir Saffari, Philip H. S. Torr
International Conference on Computer Vision (ICCV), 2011

 Slide Credit:

Relative Performance

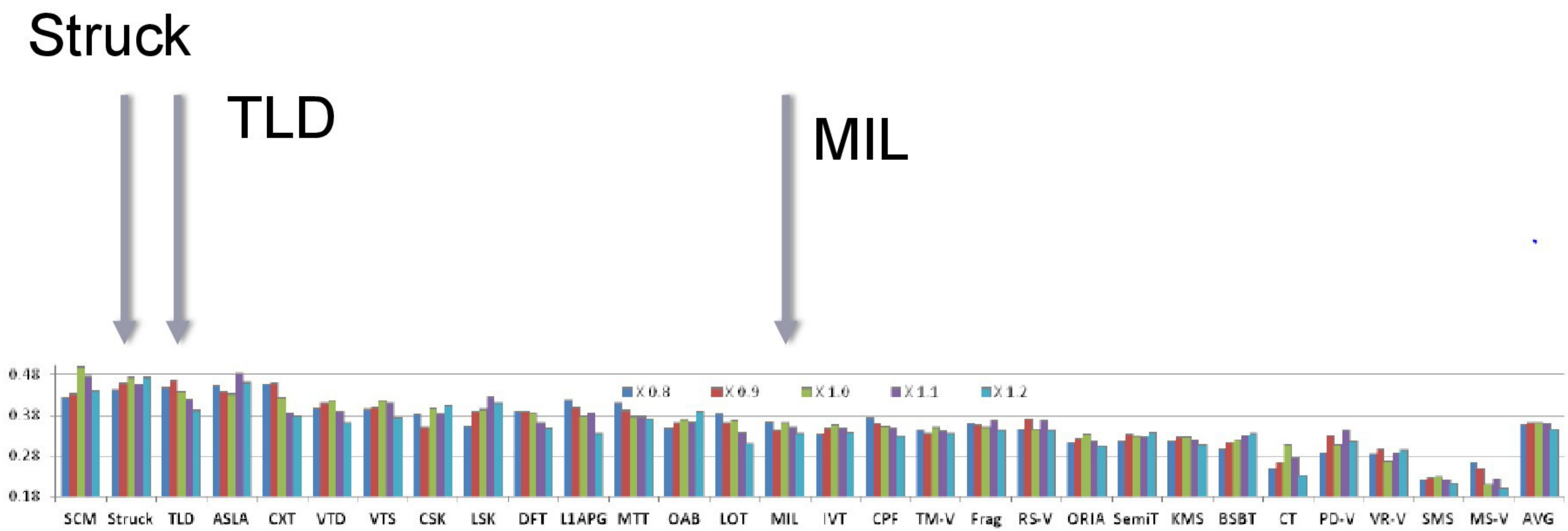


Figure 6. Performance summary for the trackers initialized with different size of bounding box. AVG (the last one) illustrates the average performance over all trackers for each scale.

Y Wu, J Lim, MH Yang "Online Object Tracking: A Benchmark", Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on

 Slide Credit: