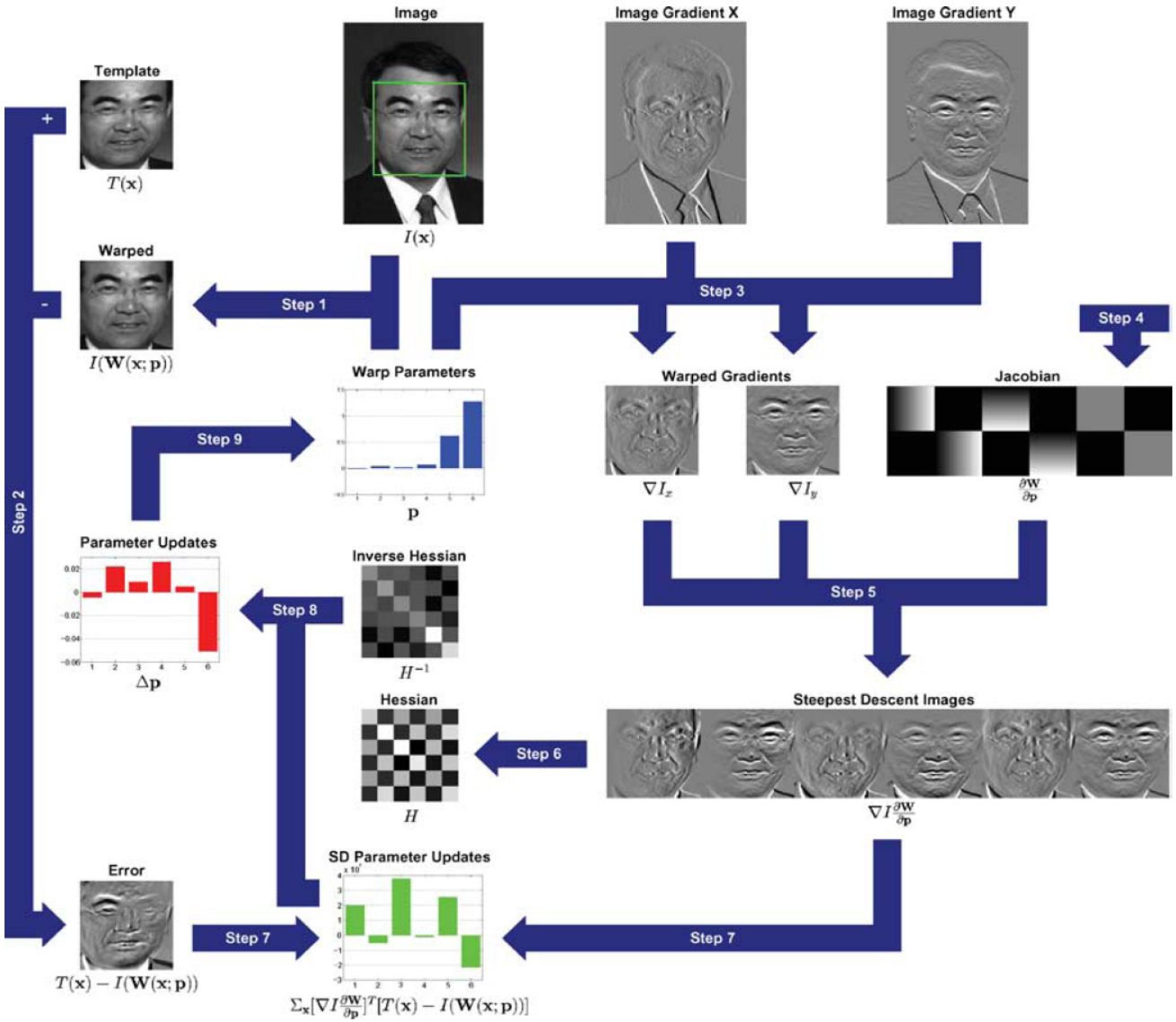
Lecture 13
Tracking

COS 429: Computer Vision

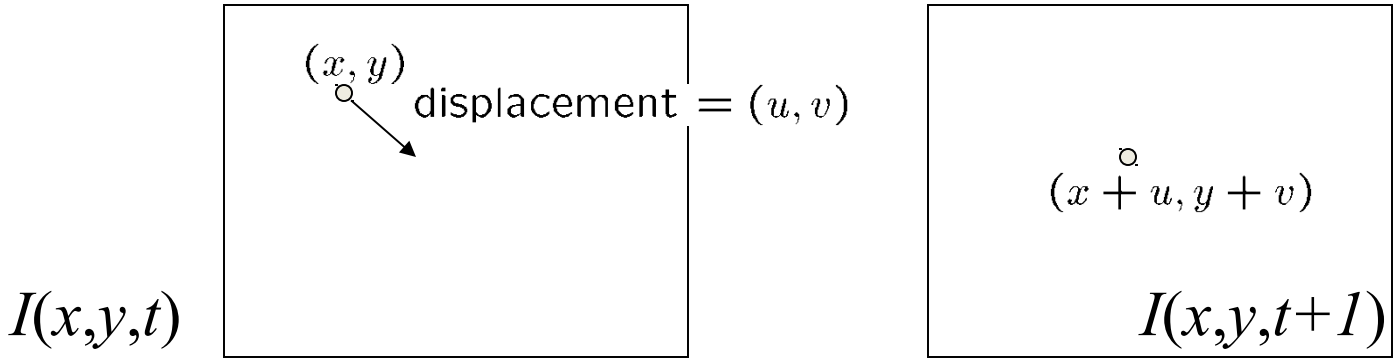
Slides credit:
Many slides adapted from James Hays, Derek Hoeim, Lana Lazebnik, Silvio Saverse, who in turn adapted slides from Steve Seitz, Rick Szeliski, Martial Hebert, Mark Pollefeys, and others

Schematic of Lucas-Kanade



Slide Credit:[Baker & Matthews, 2003]

The brightness constancy constraint



- Brightness Constancy Equation:

$$I(x,y,t) = I(x+u, y+v, t+1)$$

Take Taylor expansion of $I(x+u, y+v, t+1)$ at (x,y,t) to linearize the right side:

Image derivative along x Difference over frames

$$I(x+u, y+v, t+1) \approx I(x, y, t) + I_x \cdot u + I_y \cdot v + I_t$$

$$I(x+u, y+v, t+1) - I(x, y, t) = + I_x \cdot u + I_y \cdot v + I_t$$

Hence,

$$I_x \cdot u + I_y \cdot v + I_t \approx 0 \quad \rightarrow \nabla I \cdot \begin{bmatrix} u & v \end{bmatrix}^T + I_t = 0$$

Slide Credit:

Solving for free parameters (u,v)

- Over-constrained linear system

$$\begin{bmatrix} I_x(\mathbf{p_1}) & I_y(\mathbf{p_1}) \\ I_x(\mathbf{p_2}) & I_y(\mathbf{p_2}) \\ \vdots & \vdots \\ I_x(\mathbf{p_{25}}) & I_y(\mathbf{p_{25}}) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_t(\mathbf{p_1}) \\ I_t(\mathbf{p_2}) \\ \vdots \\ I_t(\mathbf{p_{25}}) \end{bmatrix}$$

$$\underset{25\text{x}2}{A} \quad \underset{2\text{x}1}{d} = \underset{25\text{x}1}{b}$$

Least squares solution for *d* given by $(A^T A)\ d = A^T b$

$$\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}$$

$A^T A$ $\qquad\qquad$ $A^T b$

The summations are over all pixels in the K x K window

Slide Credit:

Iterative L-K Algorithm

1) Initialize (x',y') = (x,y)

Original (x,y) position

2) Compute (u,v) by

$I_t = I(x', y', t+1) - I(x, y, t)$

$$\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}$$

2nd moment matrix for feature patch in first image

displacement

3) Shift window by (u, v): `x'=x'+u; y'=y'+v;`

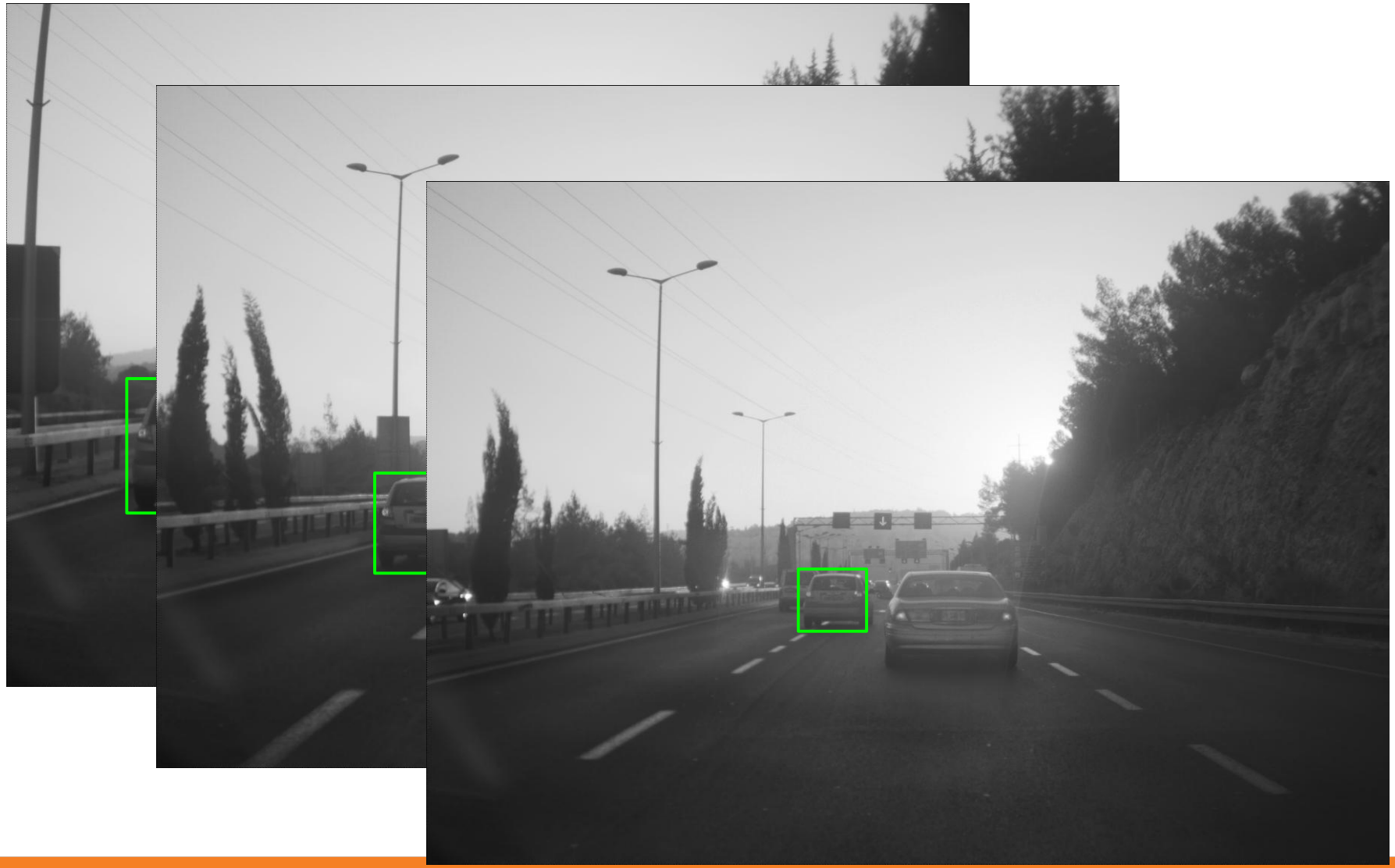
4) Recalculate I_t

5) Repeat steps 2-4 until small change

6) Use interpolation to warp by subpixel values

Slide Credit:

Object Detection and Tracking

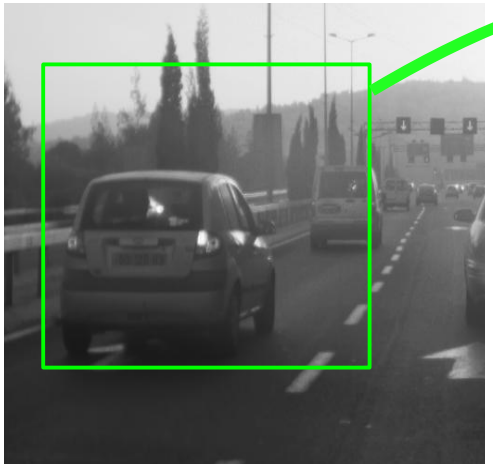


Slide Credit:

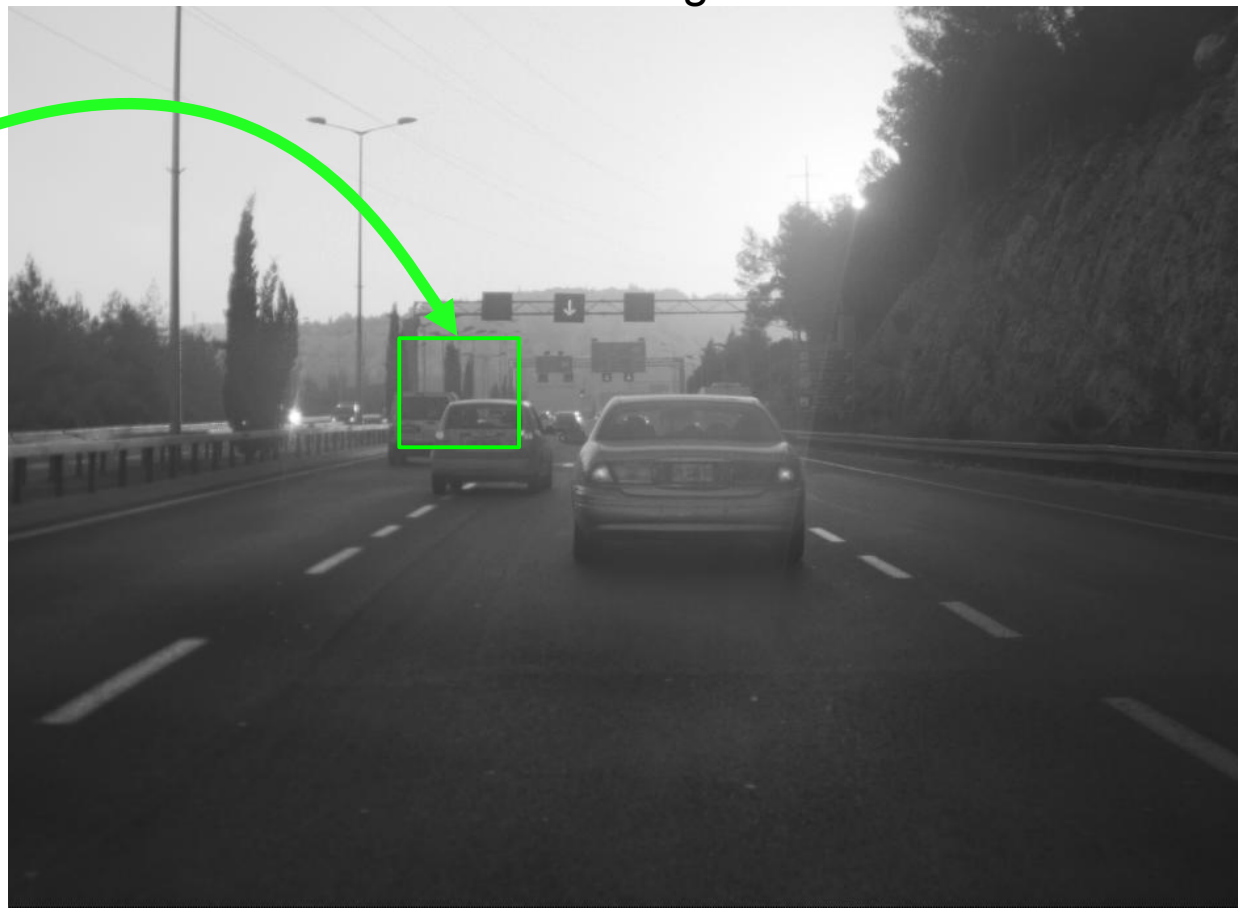
Tracking

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

Representation



Future Image Frame



Slide Credit:

Beyond Translation

Can we use Lukas-Kanade?

Yes, but need to deal with more than translation (u,v)

=> Transformations: Scale, Rigid Body, Similarty, Affine

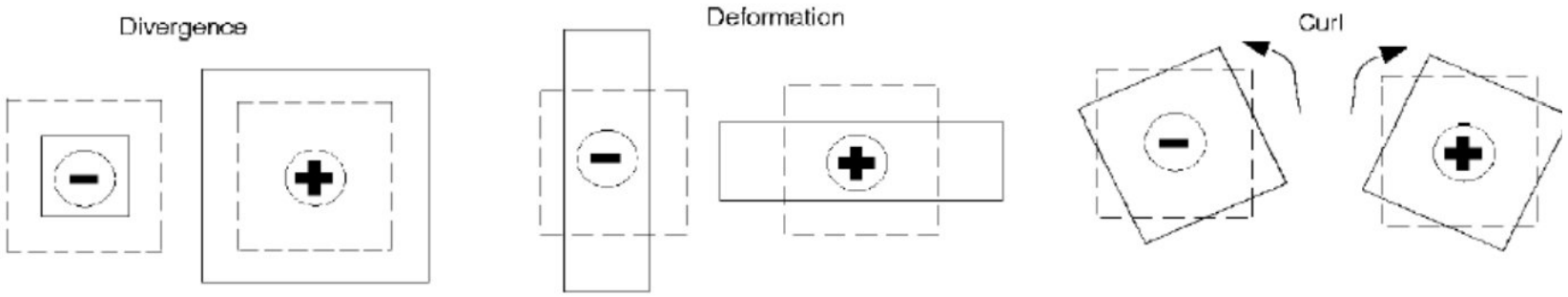
Template

Slide Credit:

Affine Motion

$$E(\mathbf{a}) = \sum_{x,y \in R} (\nabla I^T \mathbf{u}(\mathbf{x};\mathbf{a}) + I_t)^2$$

$$\mathbf{u}(\mathbf{x};\ \mathbf{a}) = \begin{bmatrix} u(\mathbf{x};\mathbf{a}) \\ v(\mathbf{x};\mathbf{a}) \end{bmatrix} = \begin{bmatrix} a_1 + a_2 x + a_3 y \\ a_4 + a_5 x + a_6 y \end{bmatrix}$$



Slide Credit: Michael Black

Affine Motion

$$\begin{bmatrix} x \\ y \end{bmatrix}^* = \begin{bmatrix} a_1 & a_2 \\ a_4 & a_5 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} a_3 \\ a_6 \end{bmatrix}$$

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}^* = \begin{bmatrix} a_1 & a_2 & a_3 \\ a_4 & a_5 & a_6 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Homogeneous coordinates

Linear transformation

Translation

Slide Credit: Michael Black

Affine Motion Optimization

$$E(\mathbf{a}) = \sum_{x,y \in R} (I_x u + I_y v + I_t)^2$$

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} a1 & a2 & a3 \\ a4 & a5 & a6 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$E(\mathbf{a}) = \sum_{x,y \in R} (I_x a_1 x + I_x a_2 y + I_x a_3 + I_y a_4 x + I_y a_5 y + I_y a_6 + I_t)^2$$

Slide Credit: Michael Black

Recall: Solving for free parameters (u,v)

- Over-constrained linear system

$$\begin{bmatrix} I_x(\mathbf{p_1}) & I_y(\mathbf{p_1}) \\ I_x(\mathbf{p_2}) & I_y(\mathbf{p_2}) \\ \vdots & \vdots \\ I_x(\mathbf{p_{25}}) & I_y(\mathbf{p_{25}}) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_t(\mathbf{p_1}) \\ I_t(\mathbf{p_2}) \\ \vdots \\ I_t(\mathbf{p_{25}}) \end{bmatrix}$$

$$\underset{25\text{x}2}{A} \quad \underset{2\text{x}1}{d} = \underset{25\text{x}1}{b}$$

Least squares solution for *d* given by $(A^T A)\ d = A^T b$

$$\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}$$

$A^T A$ $\qquad\qquad$ $A^T b$

The summations are over all pixels in the K x K window

Slide Credit:

Affine Motion Optimization

$$E(\mathbf{a}) = \sum_{x,y \in R} (I_x a_1 x + I_x a_2 y + I_x a_3 + I_y a_4 x + I_y a_5 y + I_y a_6 + I_t)^2$$

Differentiate wrt the a_i and set equal to zero.

$$\begin{bmatrix} \Sigma I_x^2 x^2 & \Sigma I_x^2 xy & \Sigma I_x^2 x & \Sigma I_x I_y x^2 & \Sigma I_x I_y xy & \Sigma I_x I_y x \\ \Sigma I_x^2 xy & \Sigma I_x^2 y^2 & \Sigma I_x^2 y & \Sigma I_x I_y xy & \Sigma I_x I_y y^2 & \Sigma I_x I_y y \\ & & & \vdots & & \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \end{bmatrix} = \begin{bmatrix} -\Sigma I_x I_t x \\ -\Sigma I_x I_t y \\ -\Sigma I_x I_t \\ -\Sigma I_y I_t x \\ -\Sigma I_y I_t y \\ -\Sigma I_y I_t \end{bmatrix}$$

Slide Credit: Michael Black

Summary

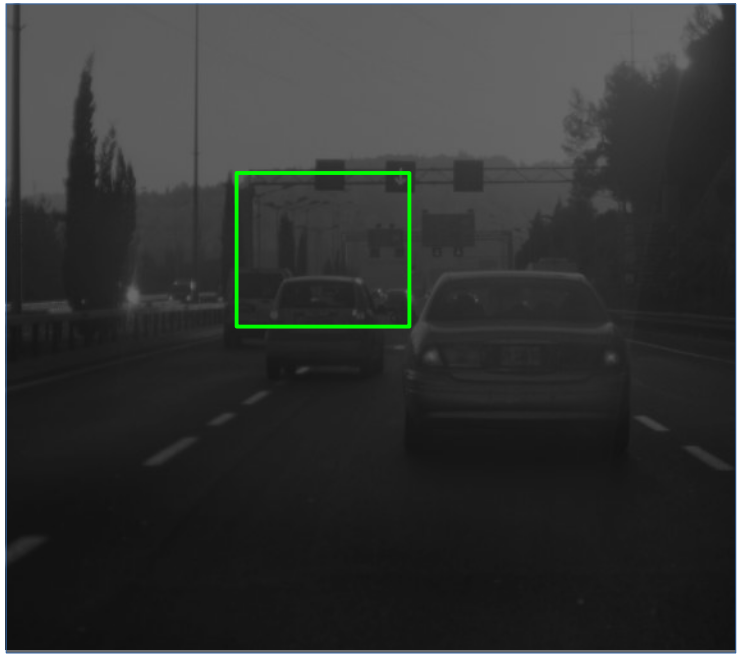
- L-K works well when:
  - Have a good initial guess
  - L2 (SSD) is a good metric
  - Can handle more degrees of freedom in motion model (scale, rotation, affine, etc.), which are too expensive for search
- But has problems with:
  - Changes in brightness
  - …

Slide Credit:

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:

Approaches to Object Tracking

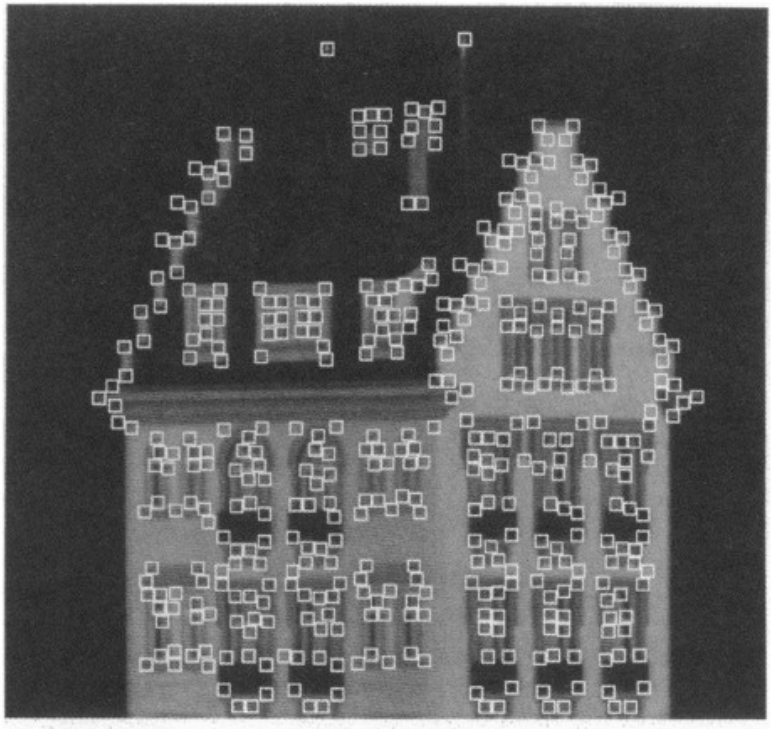
- Motion model (translation, translation+scale, affine, non-rigid, …)
- Image representation (gray/color pixel, edge image, histogram, HOG, learned features)
- Distance metric (L1, L2, normalized correlation, Chi-Squared, learned)
- Method of optimization (gradient descent, naive search, part matching)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

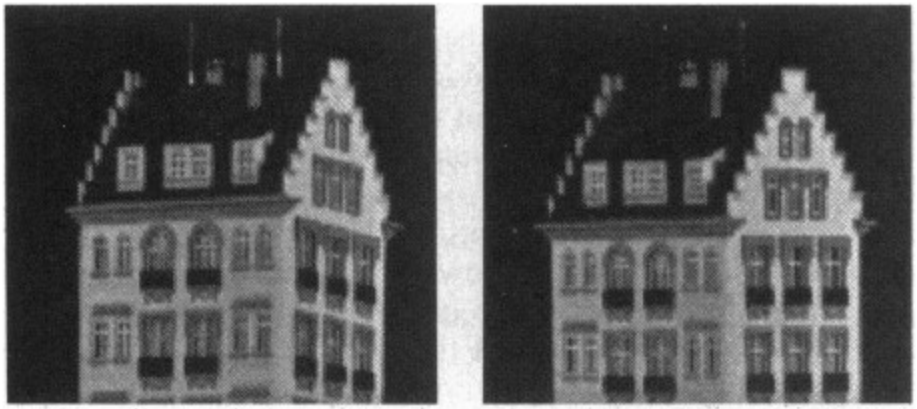
Template

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)





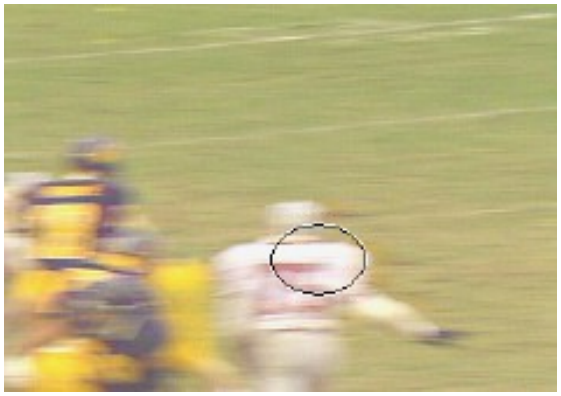
Slide Credit:

Can any of these techniques handle this?

Template

Partial occlusion

Distraction

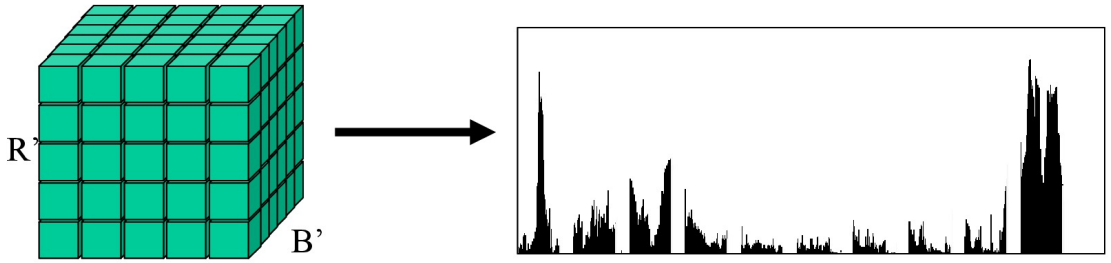


Motion blur

Slide Credit:[Baker & Matthews, 2003]

Robert Collins
CSE598C, PSU

Appearance via Color Histograms



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.

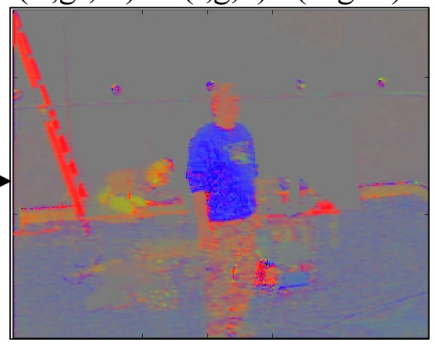
Slide Credit:

Robert Collins
CSE598C, PSU

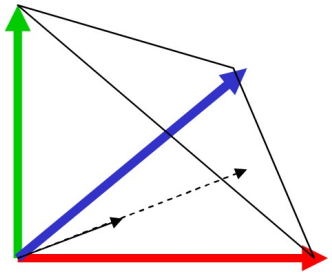
Normalized Color

(r,g,b)

(r',g',b') = (r,g,b) / (r+g+b)



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:

Robert Collins
CSE598C, PSU

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)

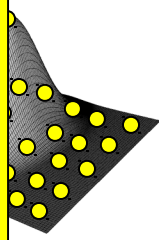
*Dorin Comanesciu, V. Ramesh and Peter Meer, "Real-time Tracking of Non-Rigid Objects using Mean Shift," IEEE Conference on Computer Vision and Pattern Recognition, Hilton Head, South Carolina, 2000 (best paper award).

Slide Credit:

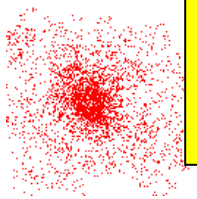
How to optimize histogram agreement?

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

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

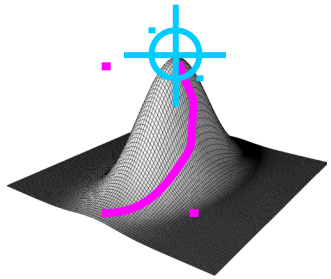


F Representation



Data

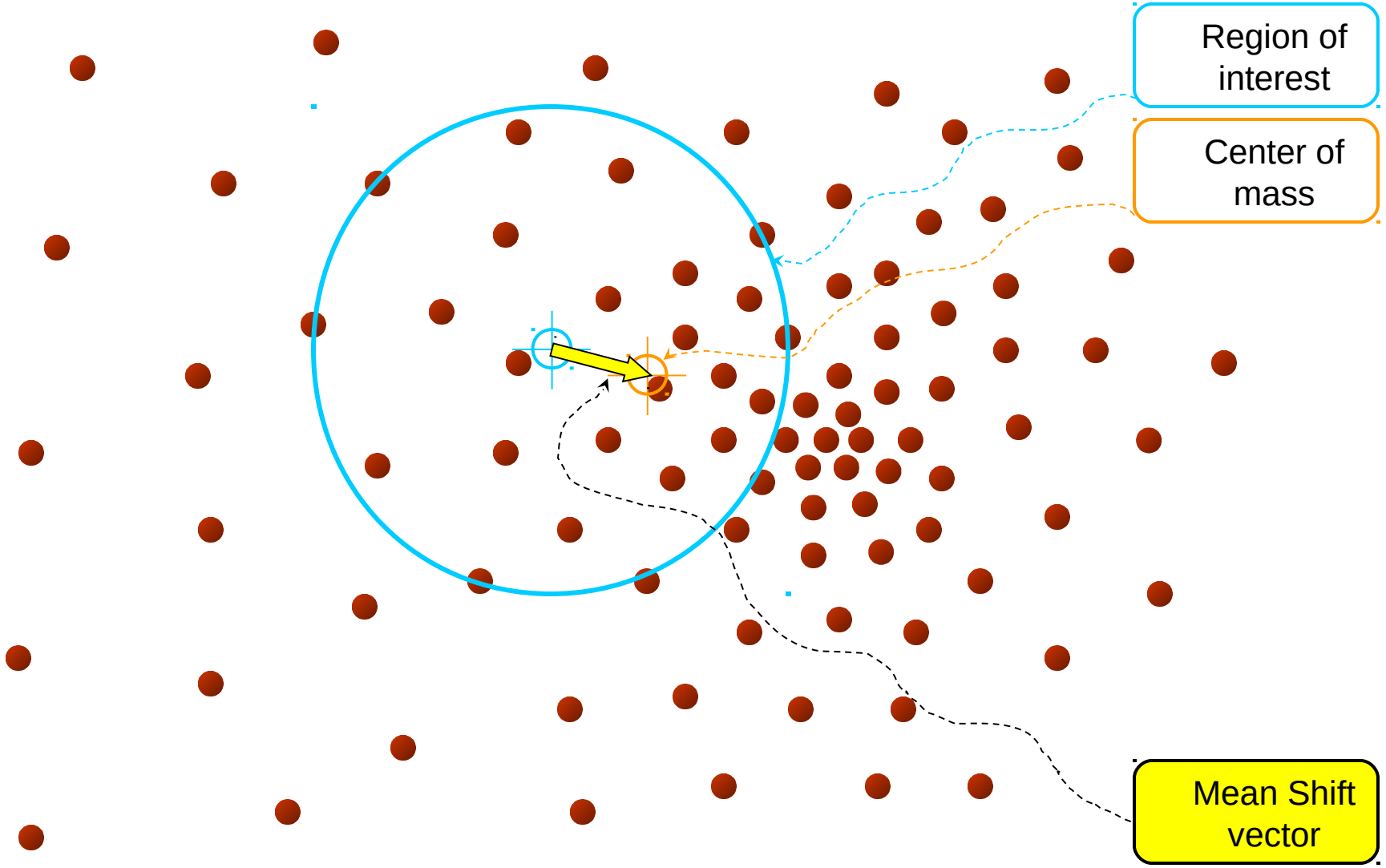
Non-parametric
Density **GRADIENT** Estimation
(Mean Shift)



PDF Analysis

Slide Credit: Y. Ukrainitz & B. Sarel

Mean Shift

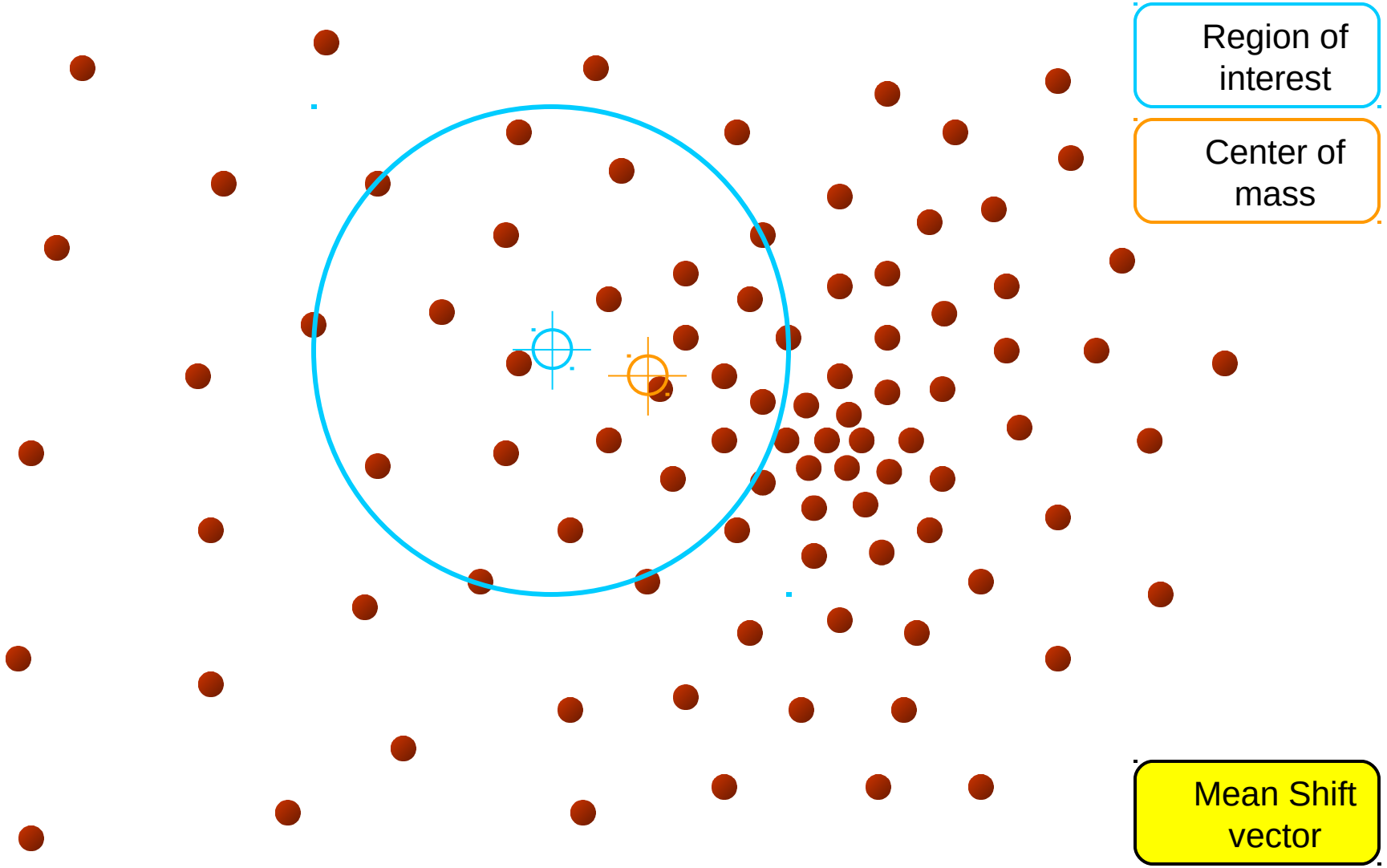


Objective : Find the densest region

Distribution of identical billiard balls

Slide Credit: Y. Ukrainitz & B. Sarel

Mean Shift

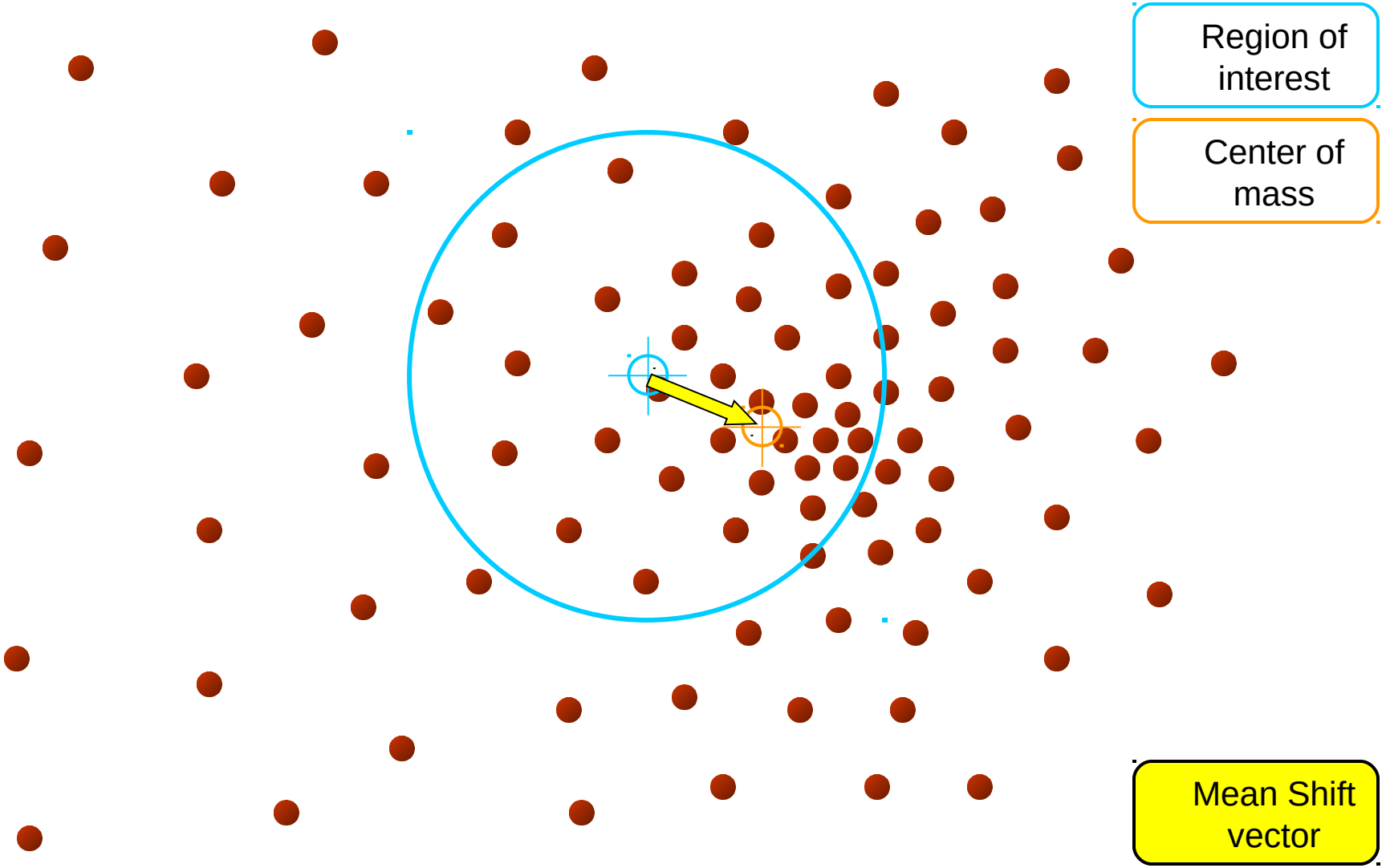


Objective : Find the densest region

Distribution of identical billiard balls

Slide Credit: Y. Ukrainitz & B. Sarel

Mean Shift

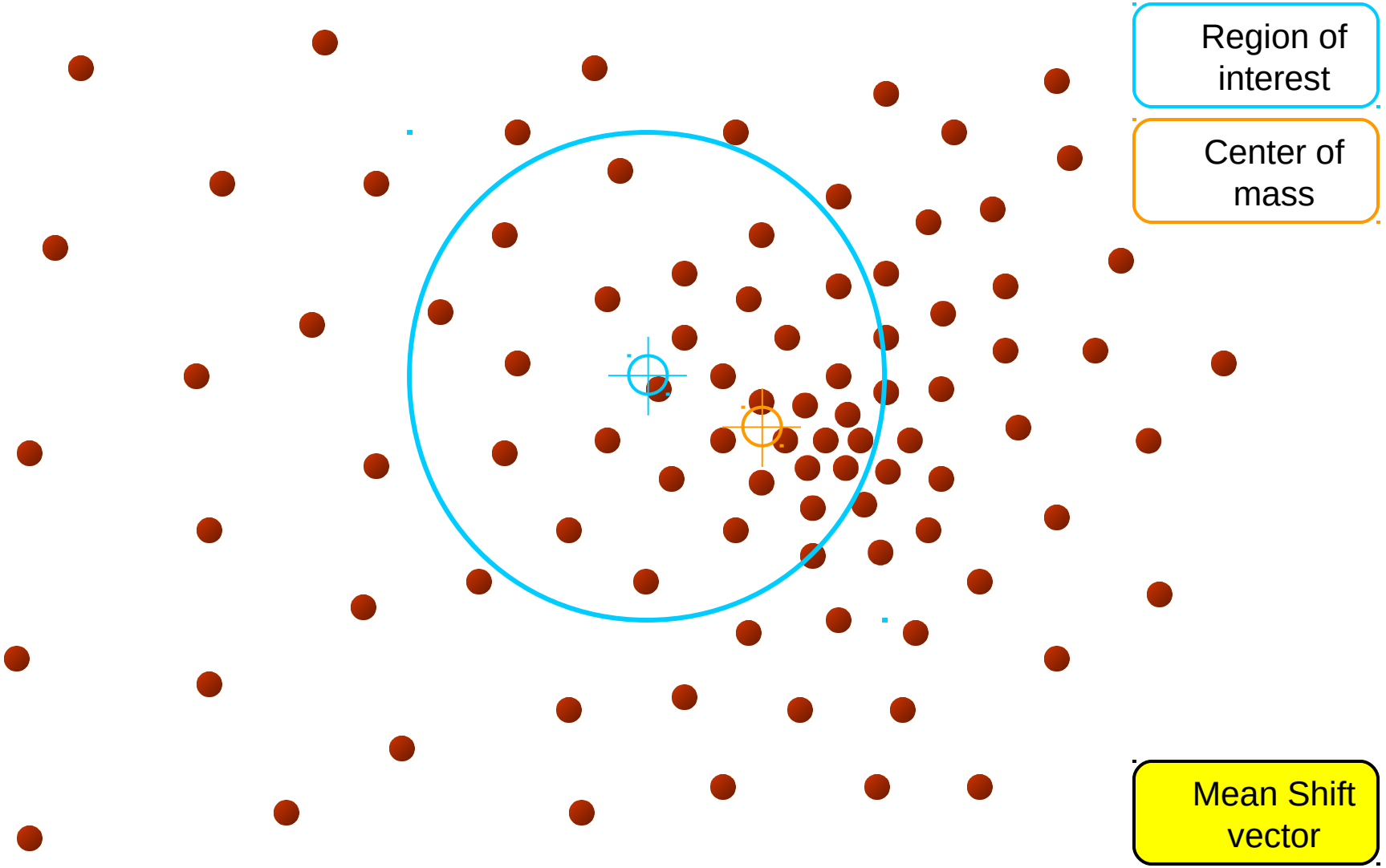


Objective : Find the densest region

Distribution of identical billiard balls

Slide Credit: Y. Ukrainitz & B. Sarel

Mean Shift

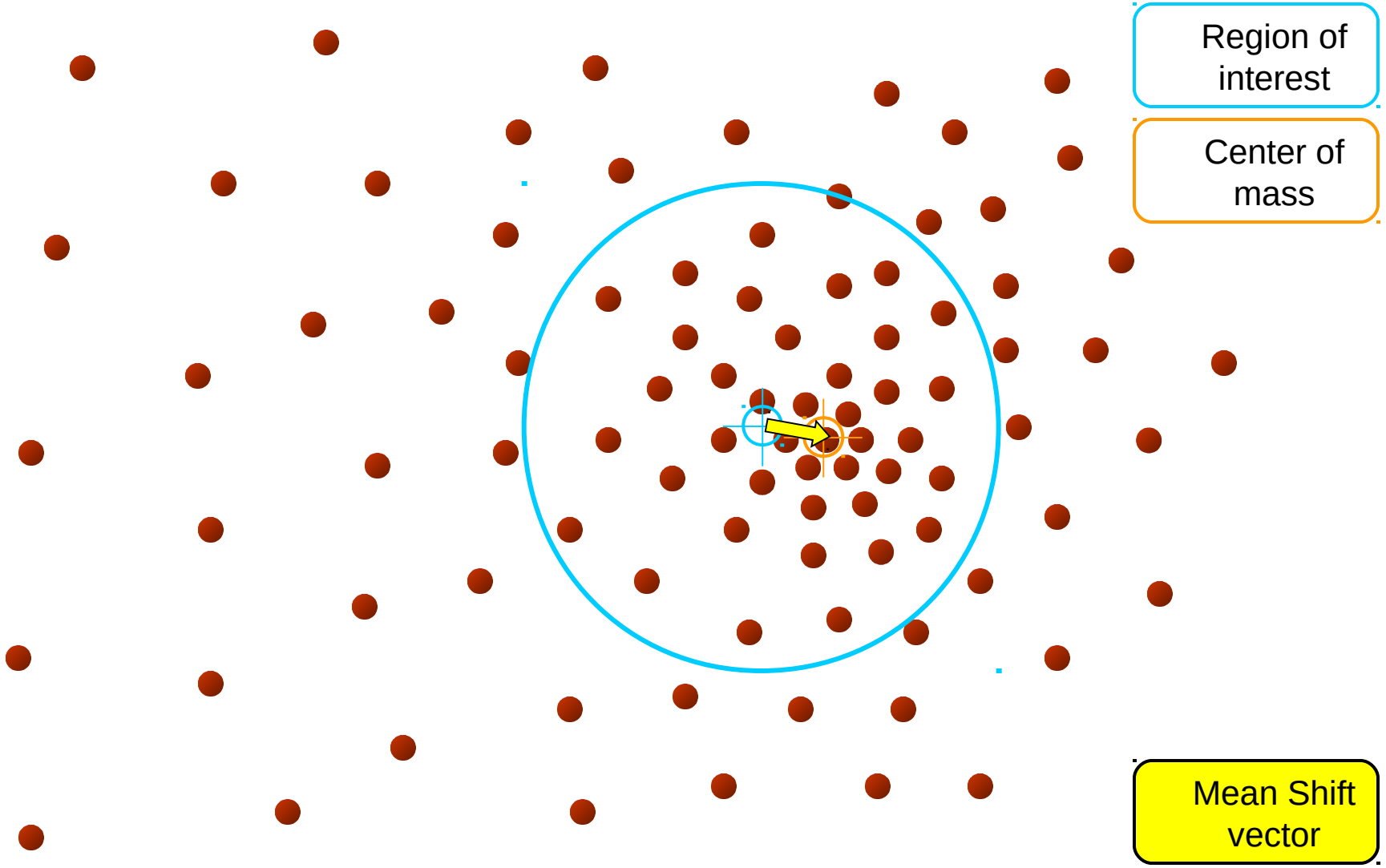


Objective : Find the densest region

Distribution of identical billiard balls

Slide Credit: Y. Ukrainitz & B. Sarel

Mean Shift

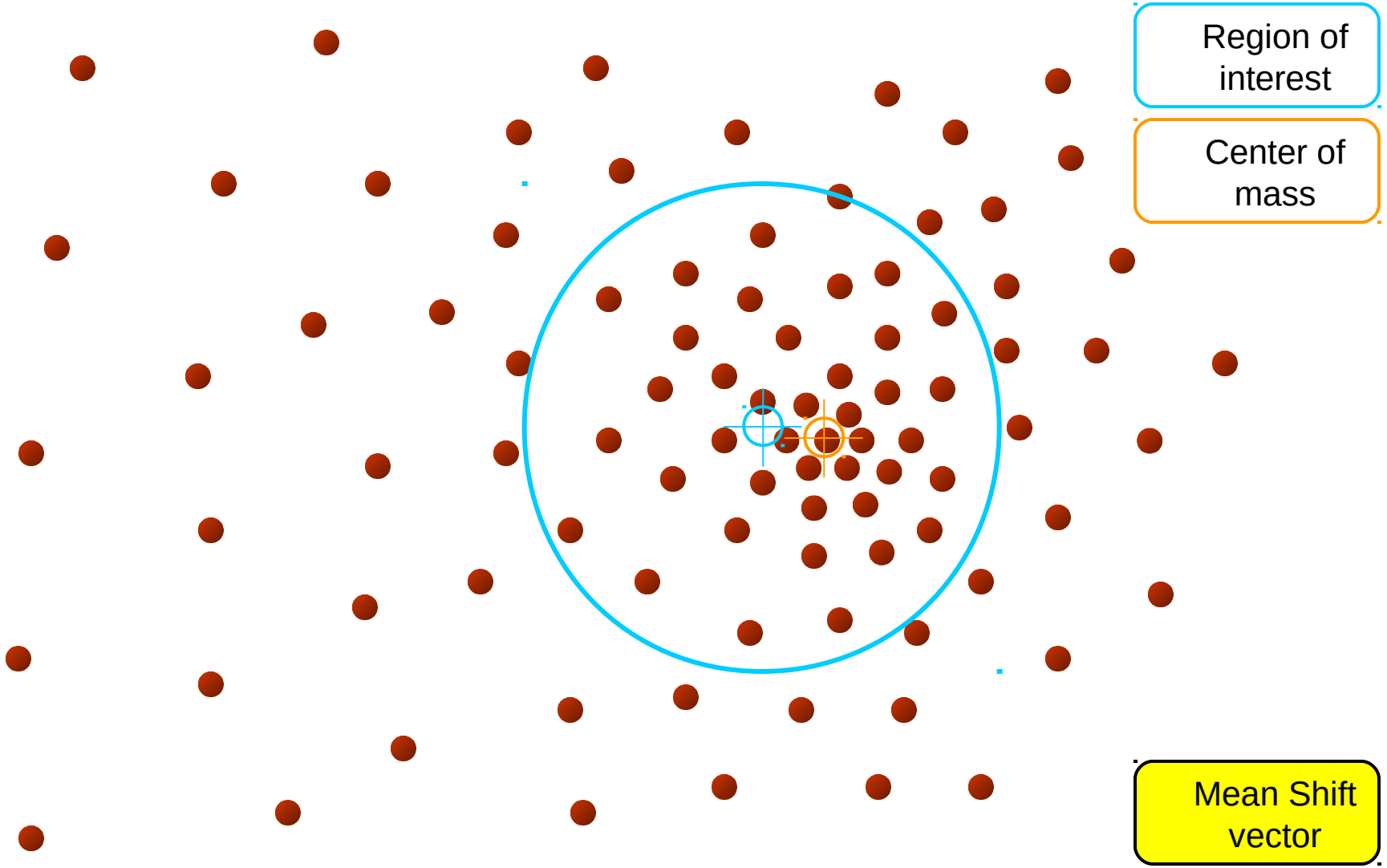


Objective : Find the densest region

Distribution of identical billiard balls

Slide Credit: Y. Ukrainitz & B. Sarel

Mean Shift

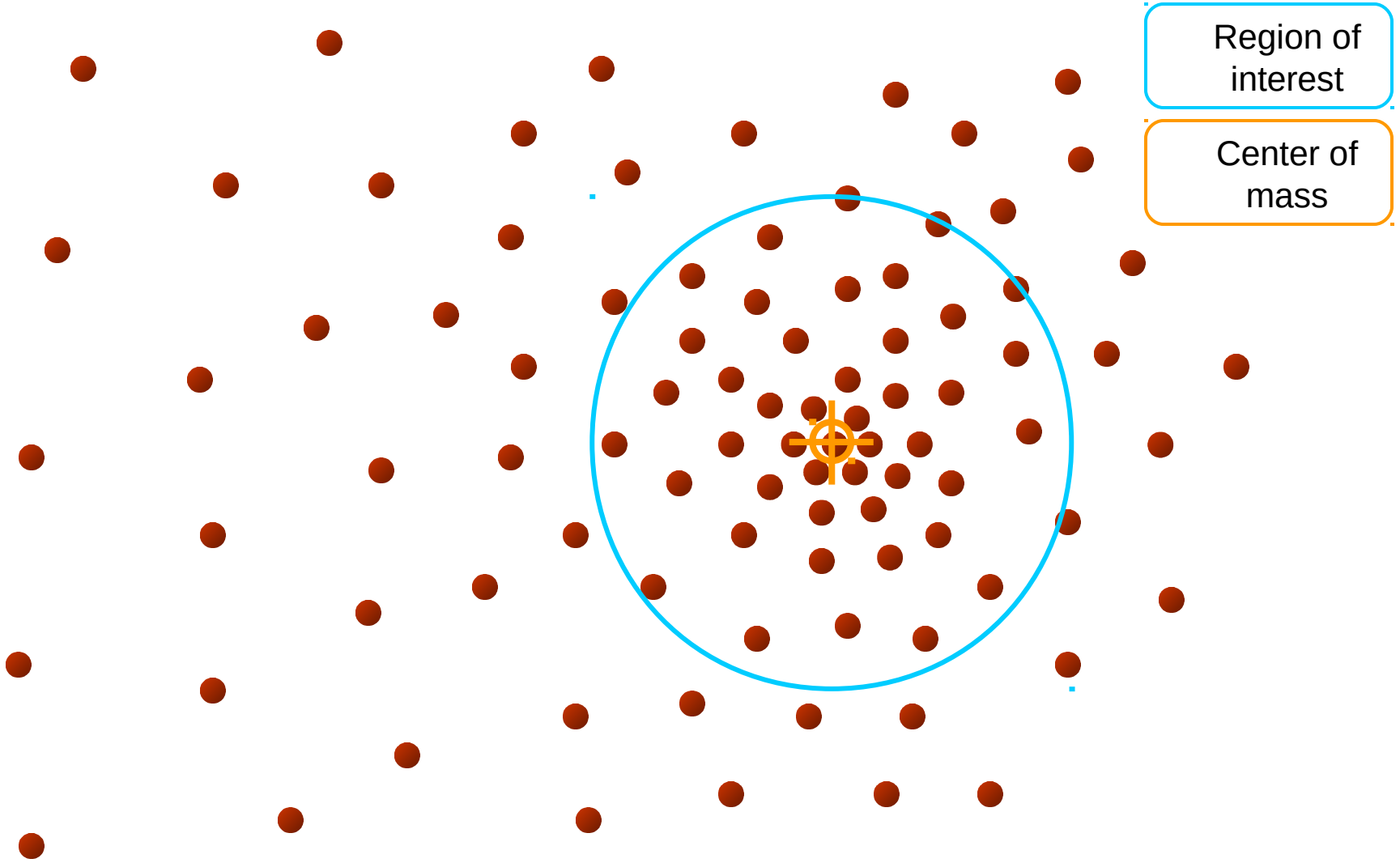


Objective : Find the densest region

Distribution of identical billiard balls

Slide Credit: Y. Ukrainitz & B. Sarel

Mean Shift



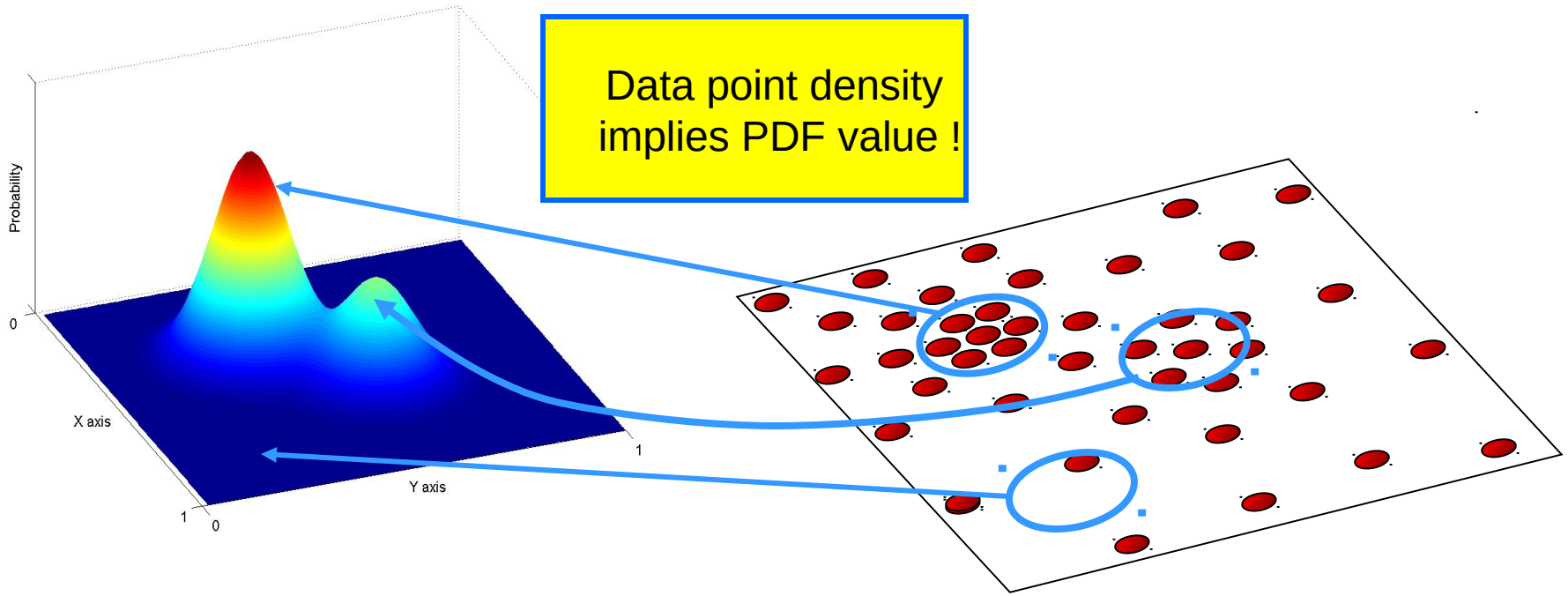
Objective : Find the densest region

Distribution of identical billiard balls

Slide Credit: Y. Ukrainitz & B. Sarel

Non-Parametric Density Estimation

Assumption : The data points are sampled from an underlying PDF



Assumed Underlying PDF **Real Data Samples**

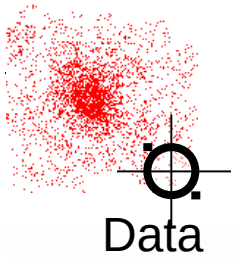
Slide Credit: **Y. Ukrainitz & B. Sarel**

Kernel Density Estimation

Various Kernels

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

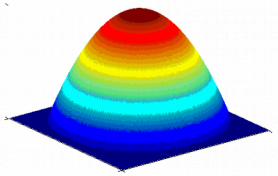
A function of some finite number of data points $x_1 \ldots x_n$



Examples:

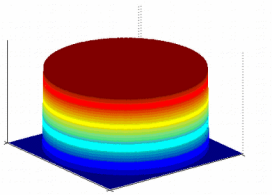
- Epanechnikov Kernel

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



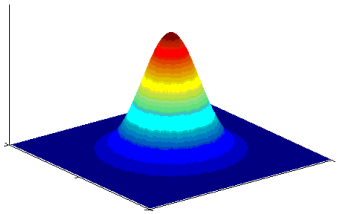
- Uniform Kernel

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



- Normal Kernel

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



Slide Credit: Y. Ukrainitz & B. Sarel

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: **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:

Robert Collins
CSE598C, PSU

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

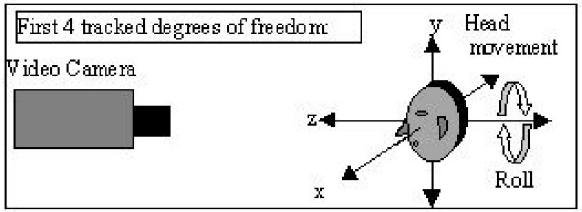
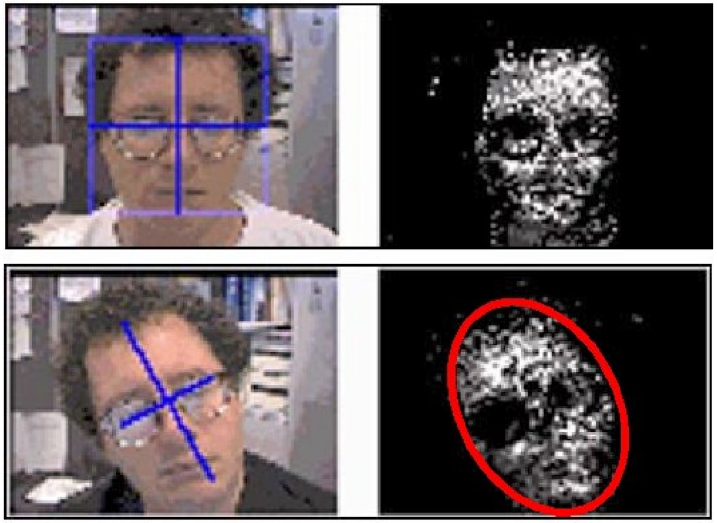


Figure 8: First four head tracked degrees of freedom: X, Y, Z location, and head roll

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.

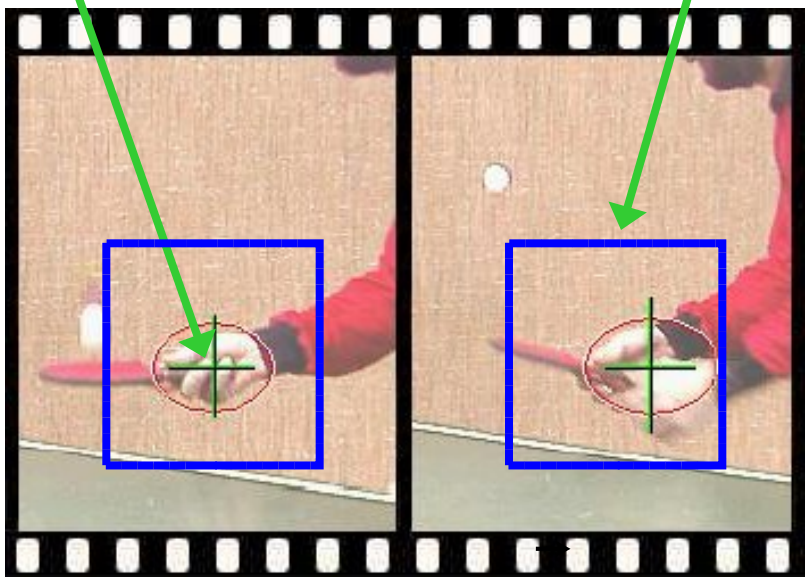
Slide Credit:

Mean-Shift Object Tracking Target Localization Algorithm

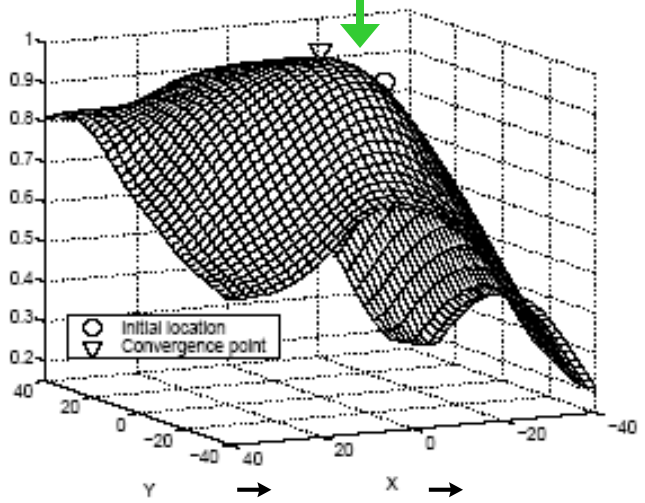
Start from the position of the model in the current frame

Search in the model's neighborhood in next frame

Find best candidate by maximizing a similarity func.



q $p(y)$



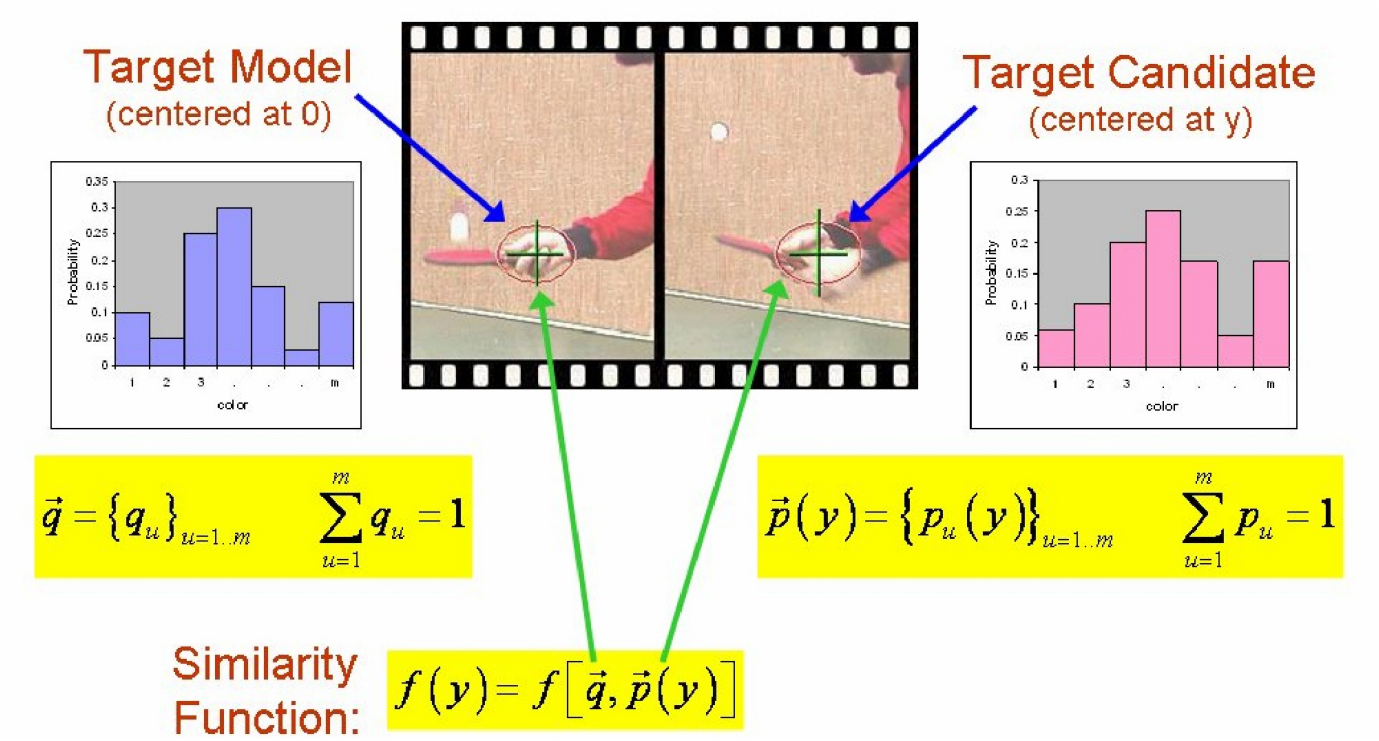
$f\left[p(y), q\right]$

Slide Credit: Y. Ukrainitz & B. Sarel

Robert Collins
CSE598C, PSU

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.

Slide Credit:

Mean-Shift Object Tracking

Results

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

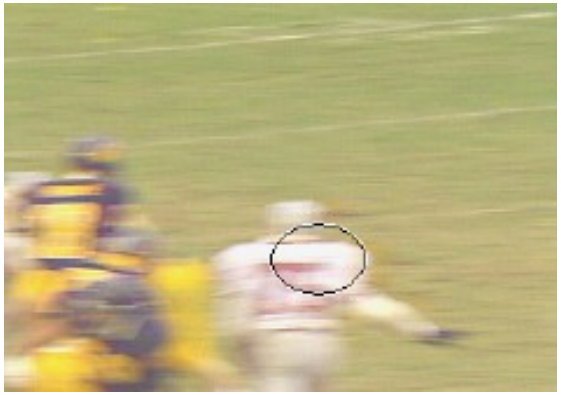
Slide Credit:

Mean-Shift Object Tracking

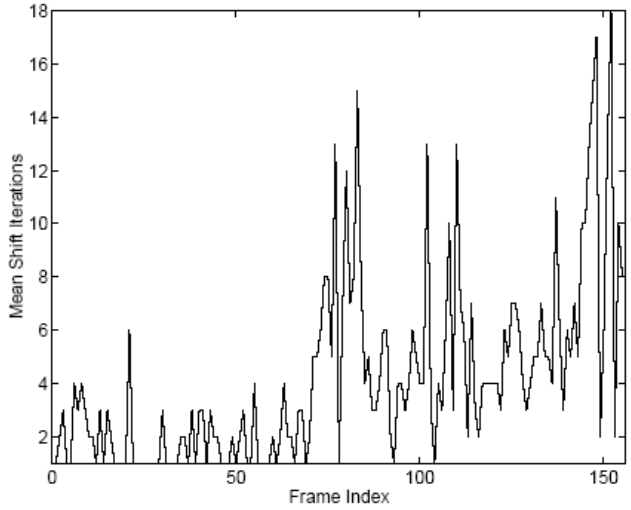
Results

Partial occlusion

Distraction

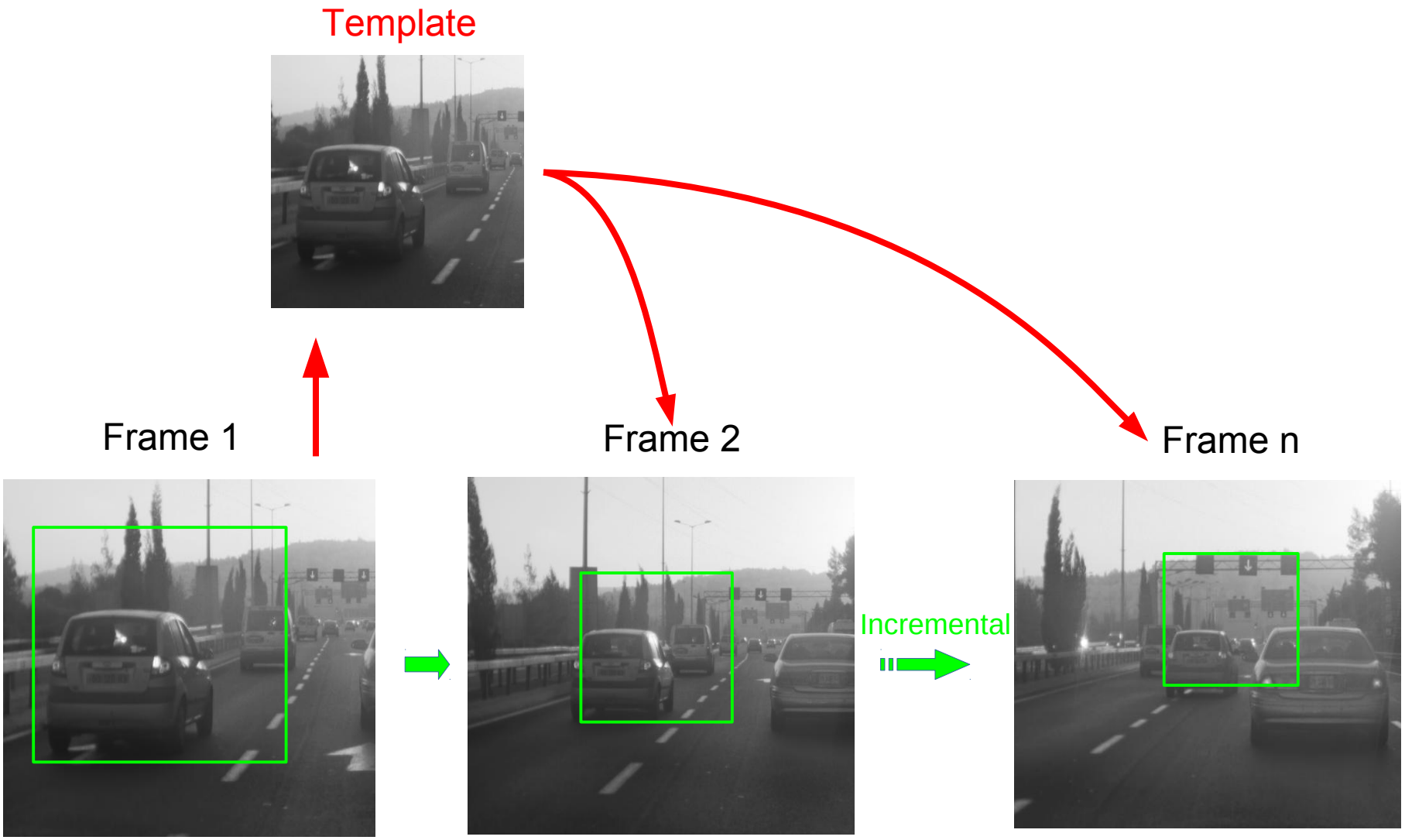


Motion blur



Slide Credit: Y. Ukrainitz & B. Sarel

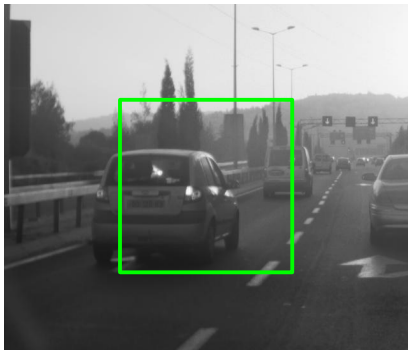
Incremental Tracking vs Template Tracking



Slide Credit:

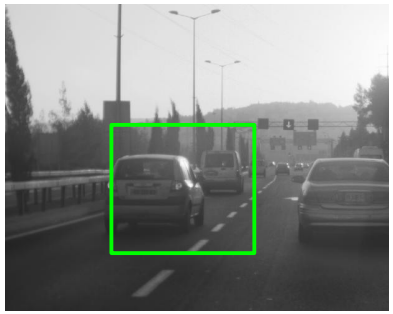
Tracking a Sequence

Original Template
(t_0)



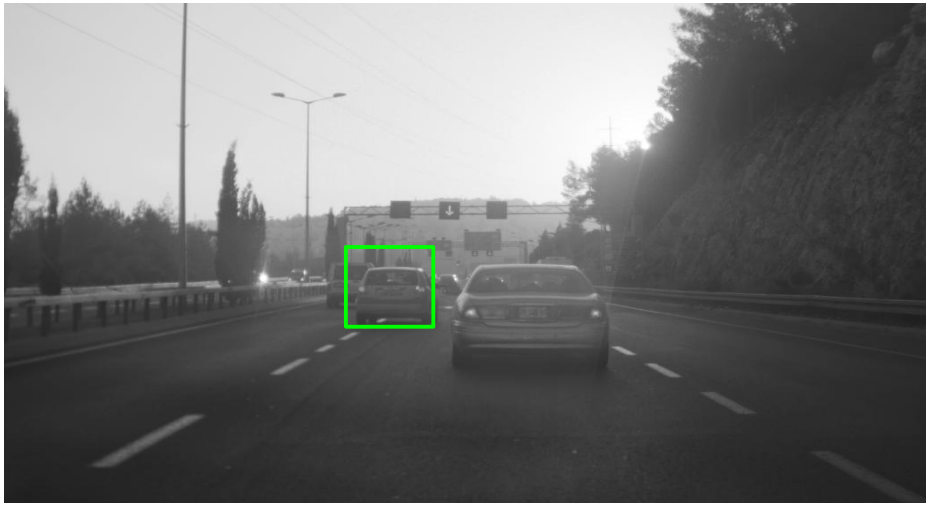
. . .

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:

Tracking by Detection

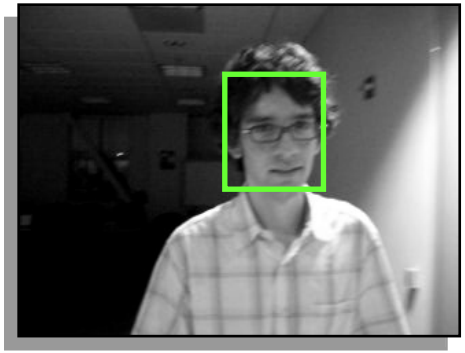
- Focus on appearance model
- Borrow techniques from obj. detection
  - Slide a discriminative classifier around image
- **Adaptive** appearance model

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

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

Tracking by Detection

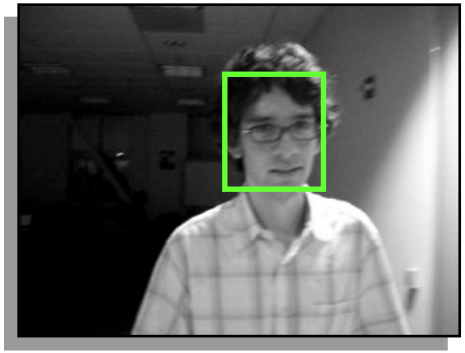
- First frame is labeled



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

Tracking by Detection

- First frame is labeled

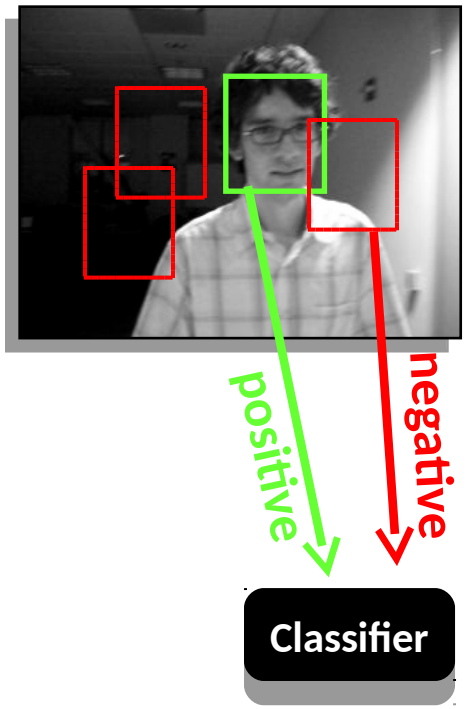


Classifier ← Online classifier (i.e. Online AdaBoost)

Slide Credit: Babenko, Yang, Belongie

Tracking by Detection

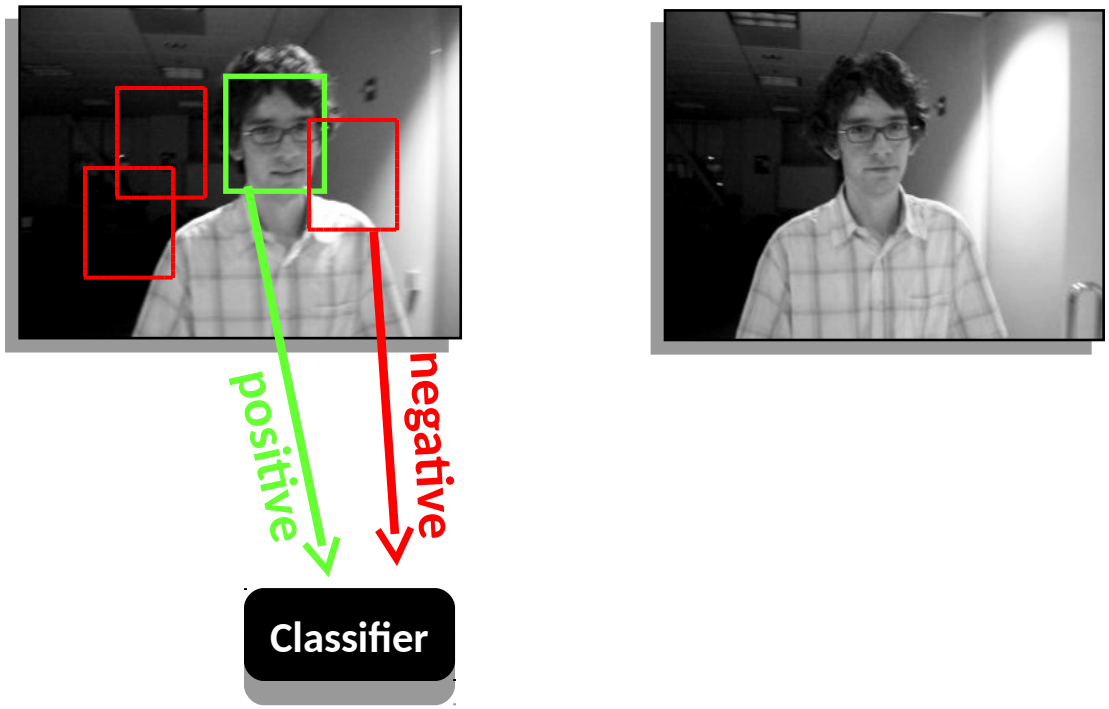
- Grab one **positive** patch, and some **negative** patch, and train/update the model.



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

Tracking by Detection

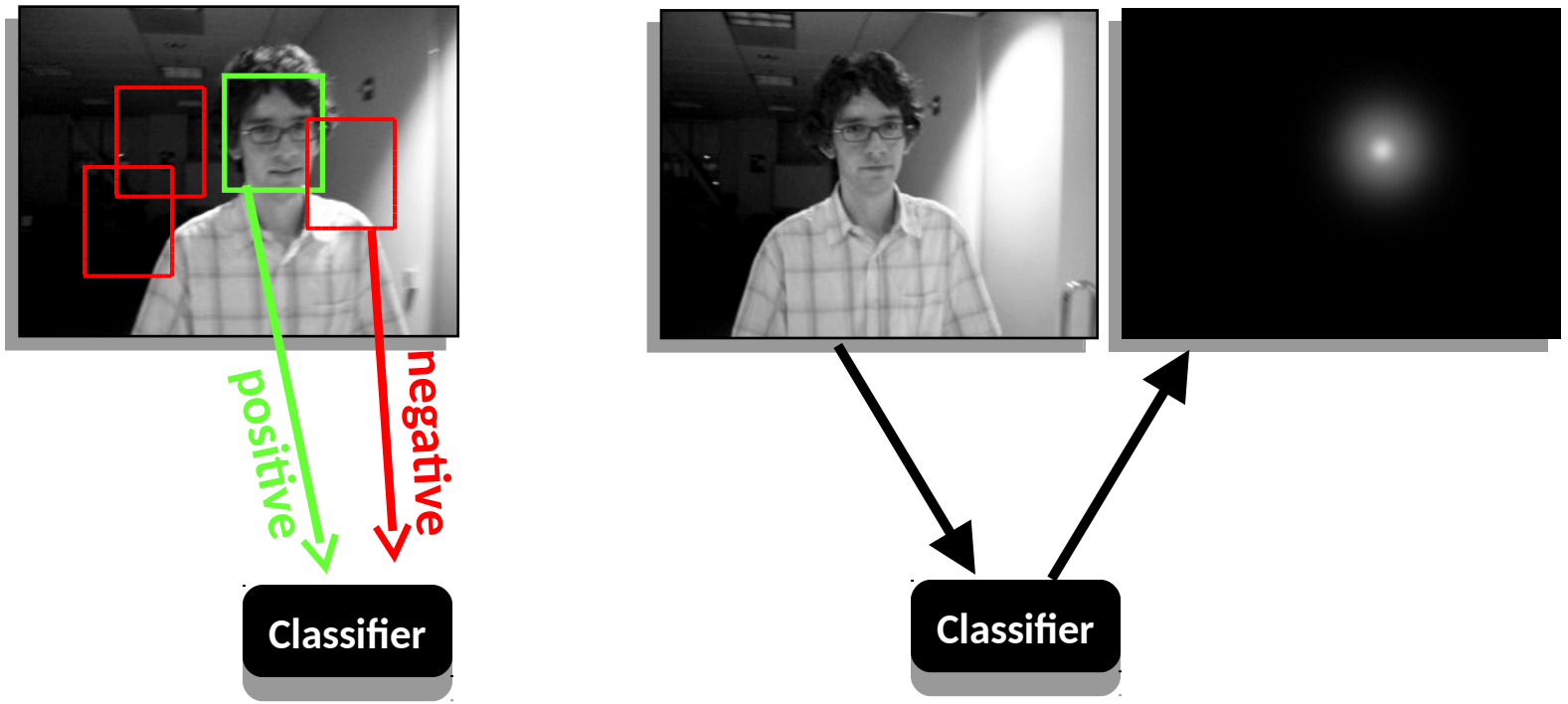
- Get next frame



Slide Credit: Babenko, Yang, Belongie

Tracking by Detection

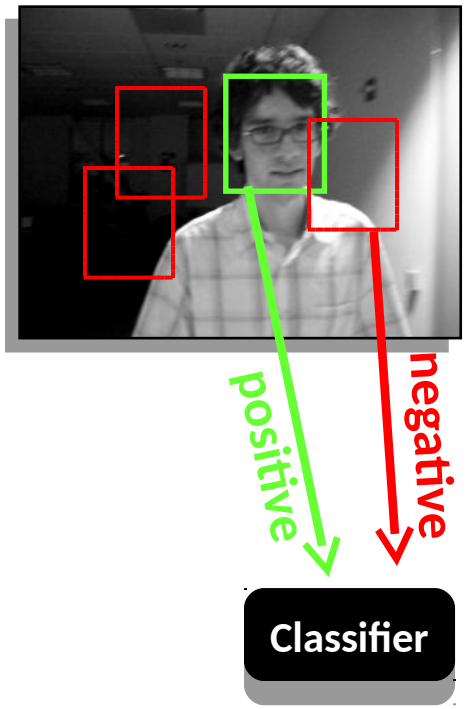
- Evaluate classifier in some search window

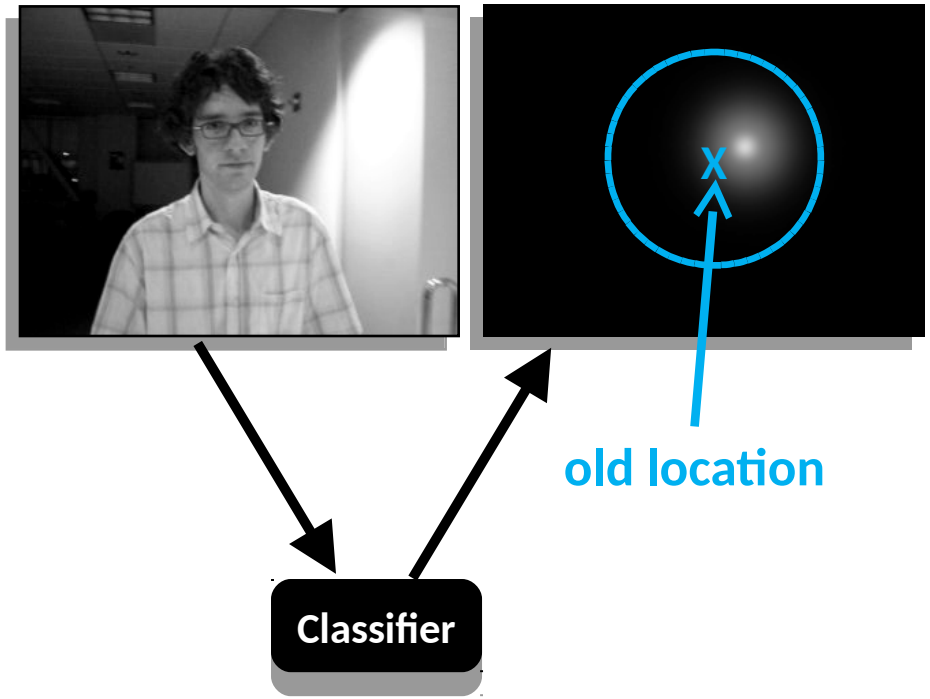


Slide Credit: Babenko, Yang, Belongie

Tracking by Detection

- Evaluate classifier in some search window

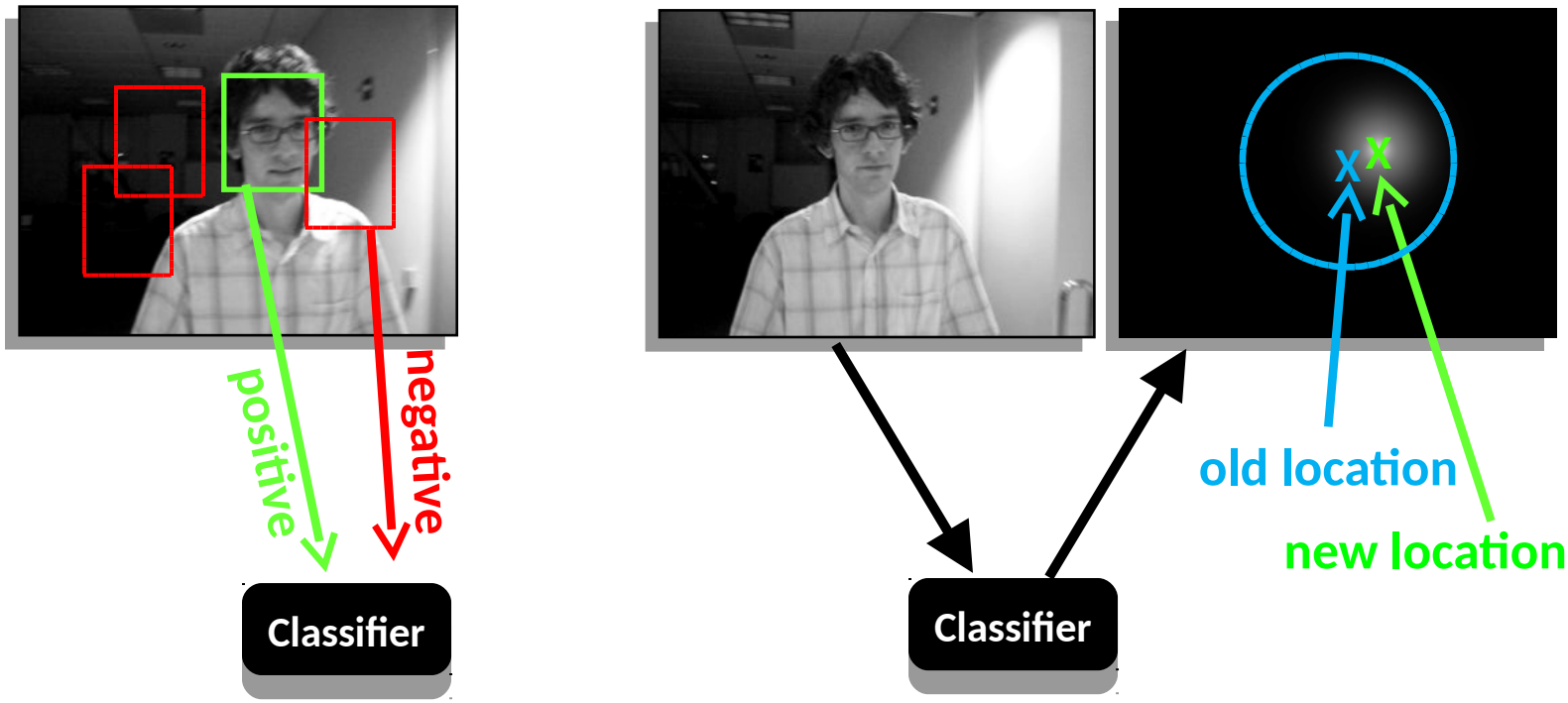




Slide Credit: Babenko, Yang, Belongie

Tracking by Detection

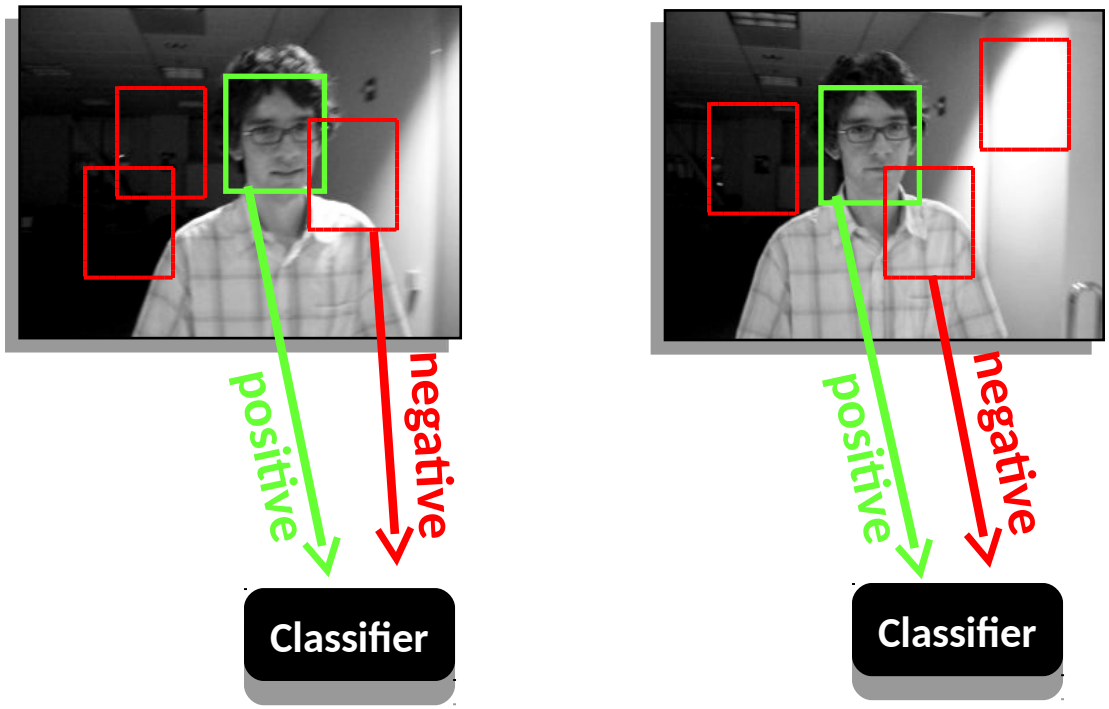
- Find max response



Slide Credit: Babenko, Yang, Belongie

Tracking by Detection

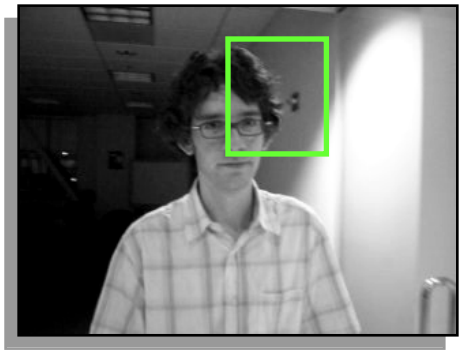
- Repeat…



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

Problems with Adaptive Appearance Models

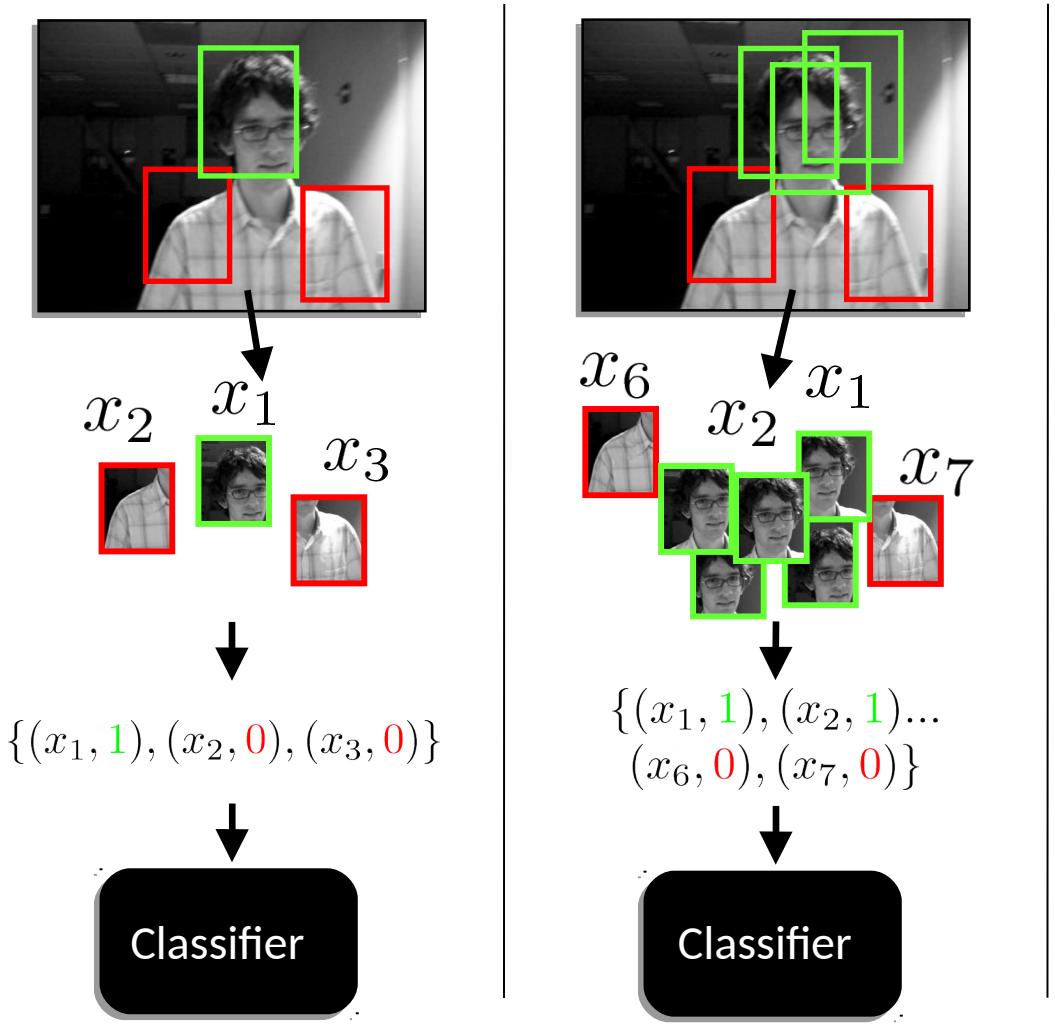
- What if classifier is a bit off?
  - Tracker starts to drift



- How to choose training examples?

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

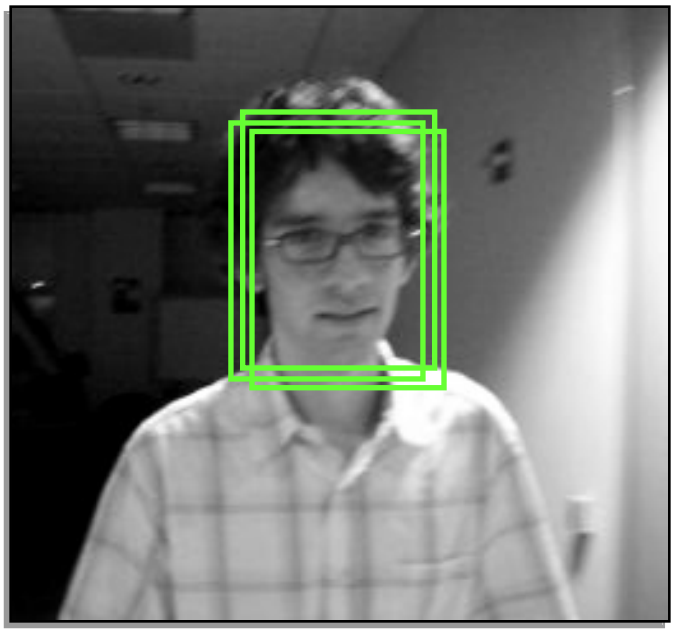
How to Get Training Examples



Slide Credit: Babenko, Yang, Belongie

Object Detection

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



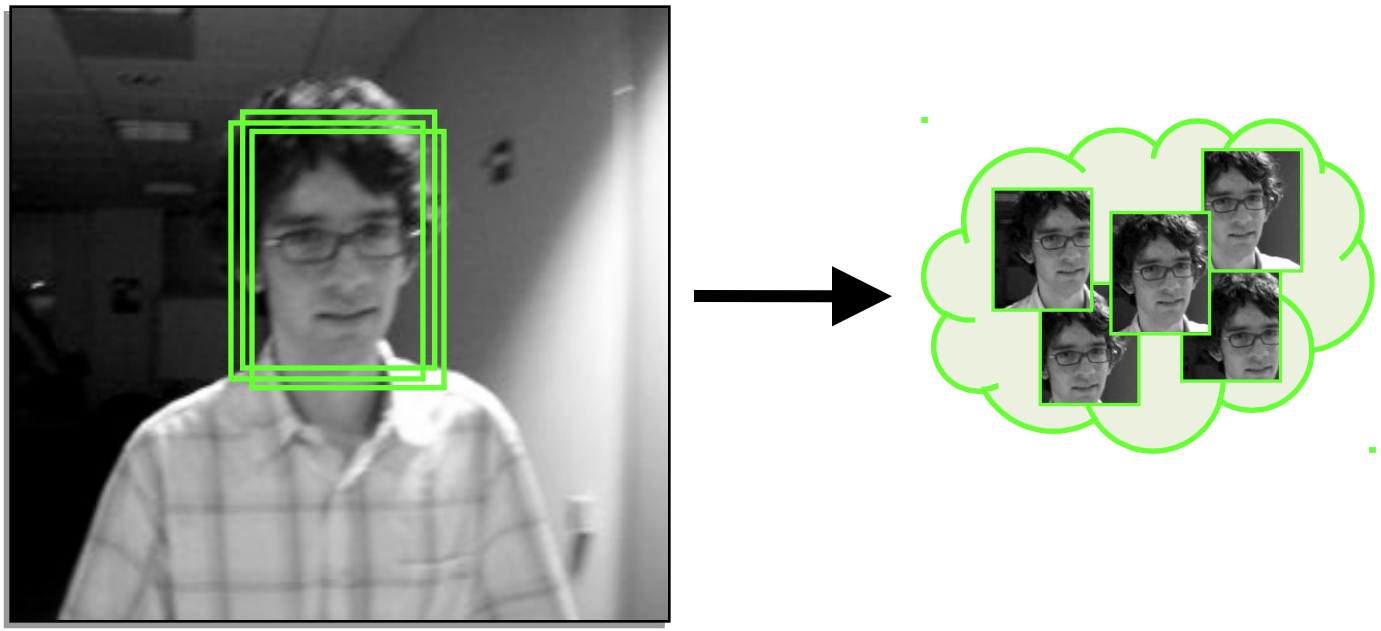
[Viola et al. ‘05]

Slide Credit: Babenko, Yang, Belongie

MIL for Object Detection

Visual Tracking with Online Multiple Instance Learning (2009)
Boris Babenko, Ming-Hsuan Yang, Serge Belongie

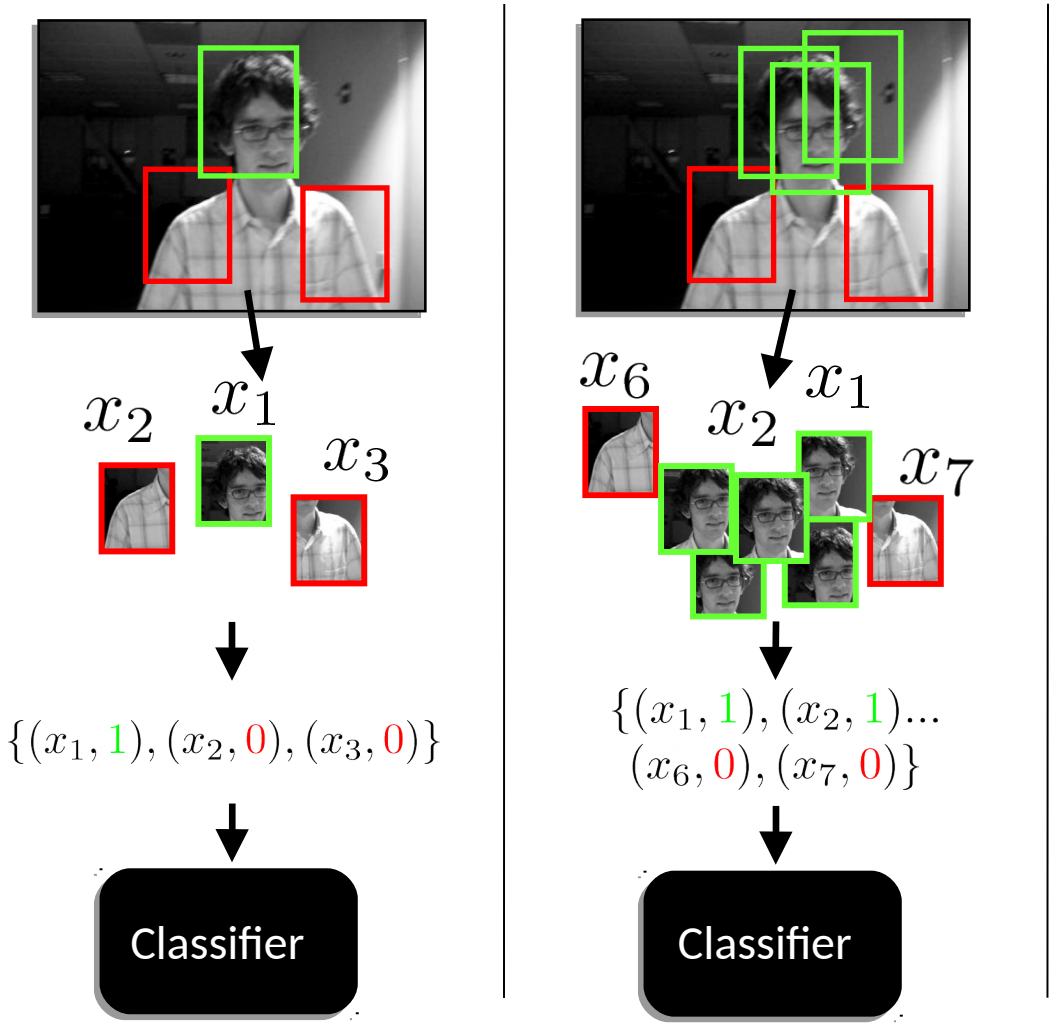
- MIL Solution:
  - Take all of these patches, put into **positive bag**
  - At least one patch in bag is “correct”



[Viola et al. ‘05]

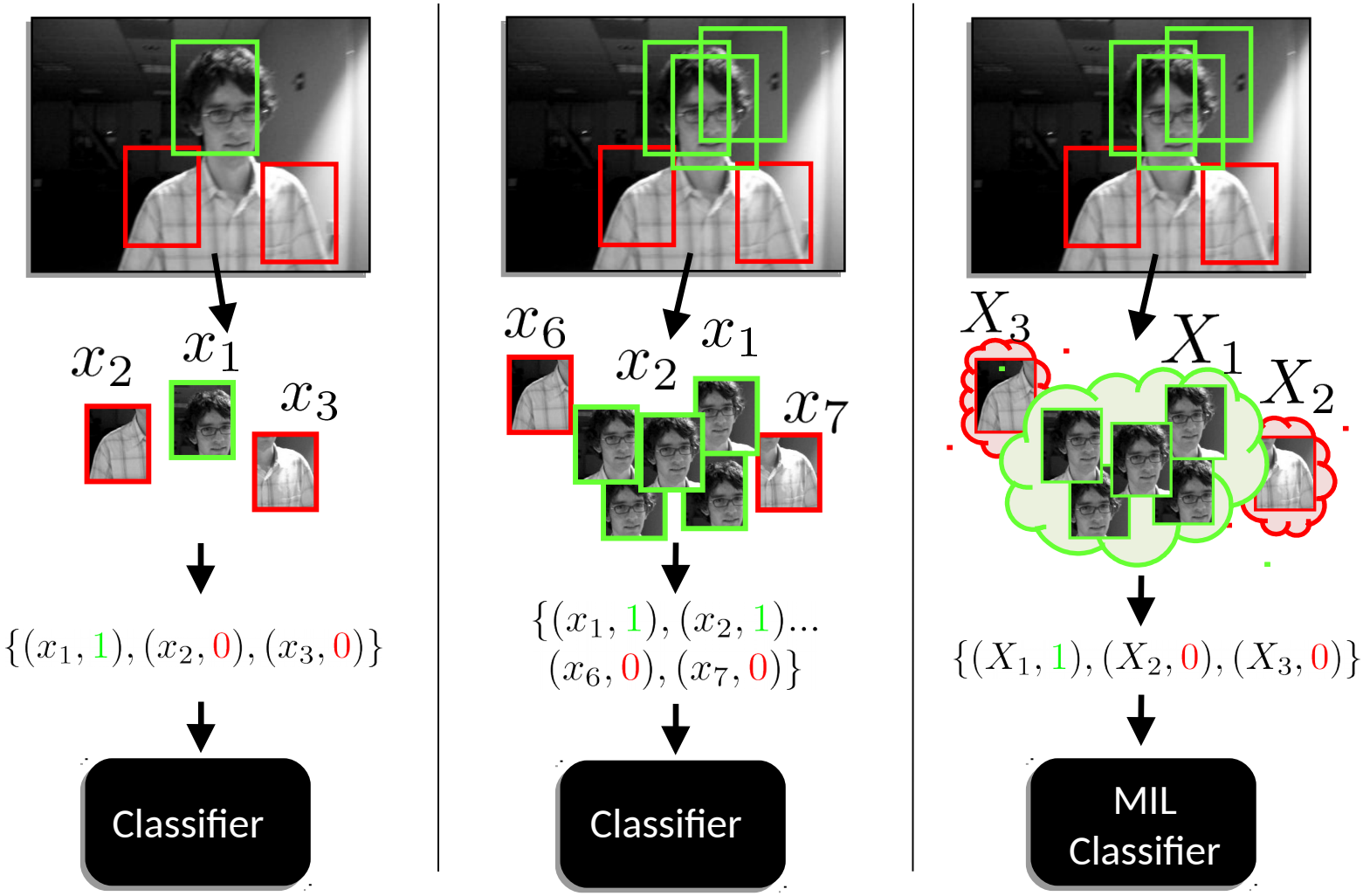
Slide Credit: **Babenko, Yang, Belongie**

How to Get Training Examples



Slide Credit: Babenko, Yang, Belongie

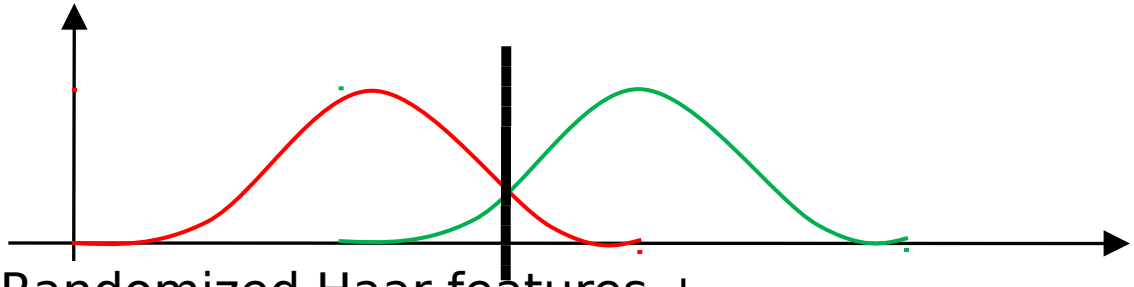
How to Get Training Examples



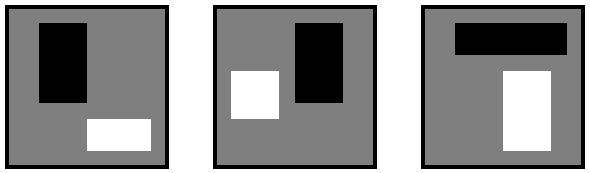
Slide Credit: **Babenko, Yang, Belongie**

MILTrack

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



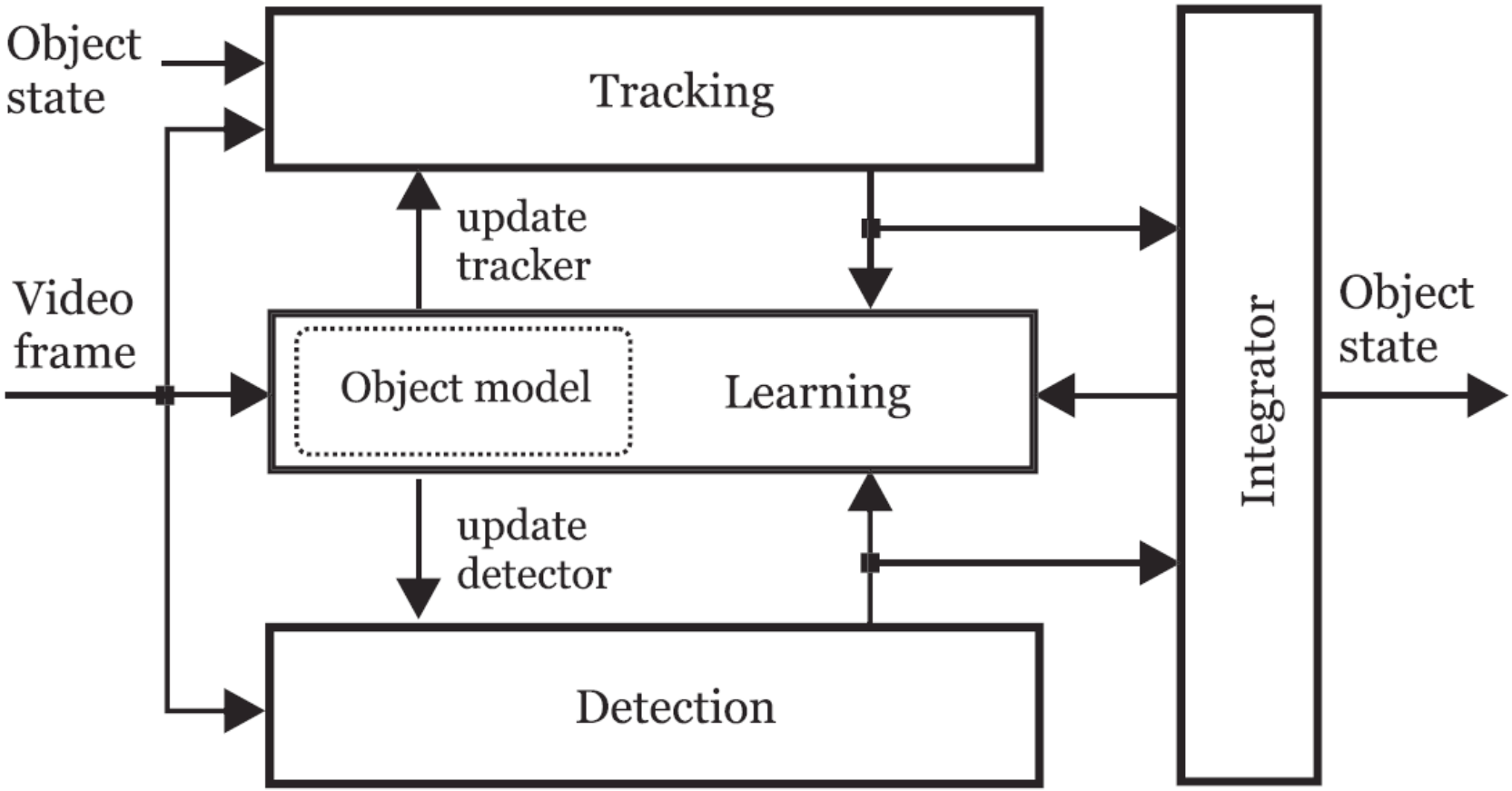
  - Randomized Haar features +



  - 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

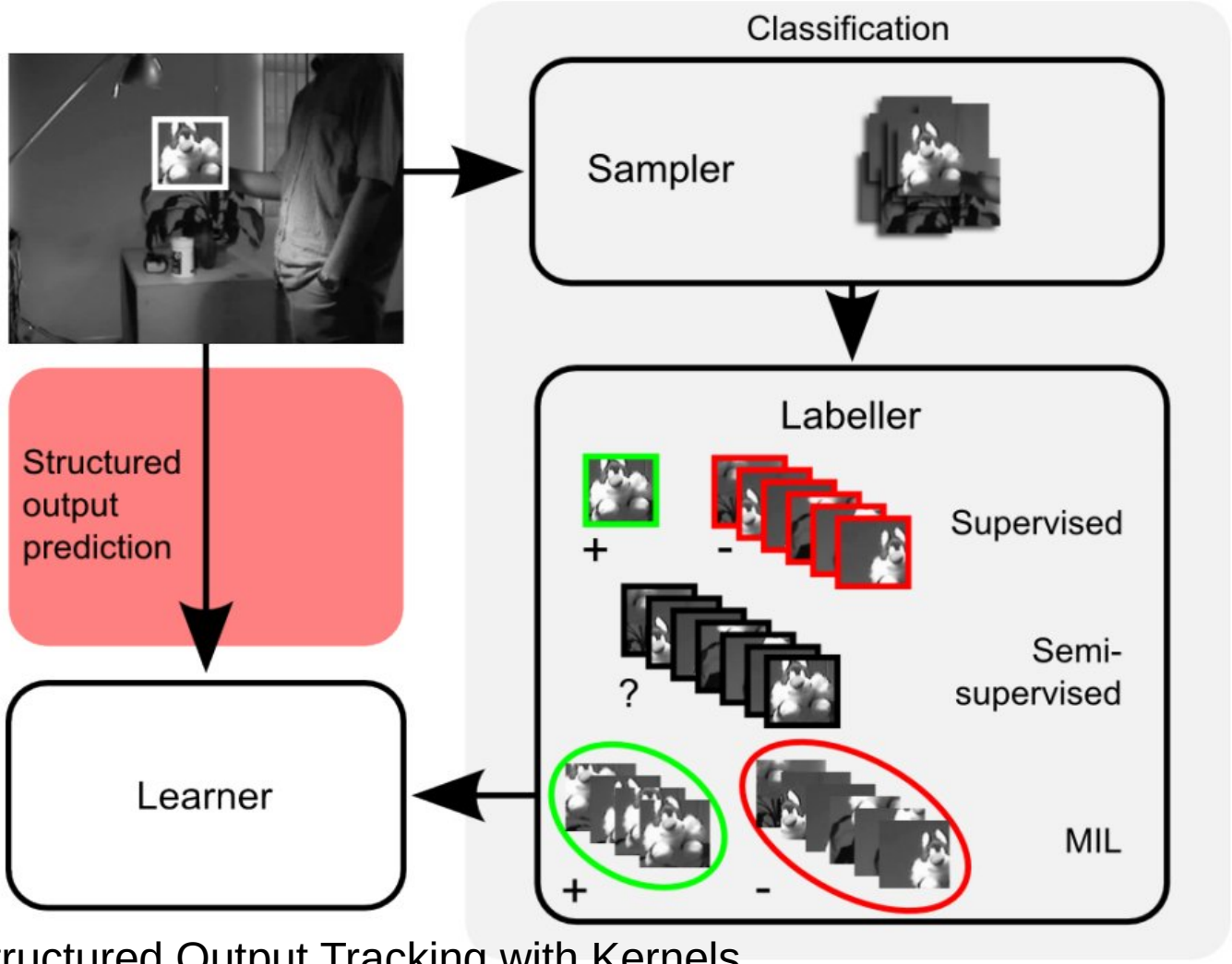
Both of the experts make errors themselves

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

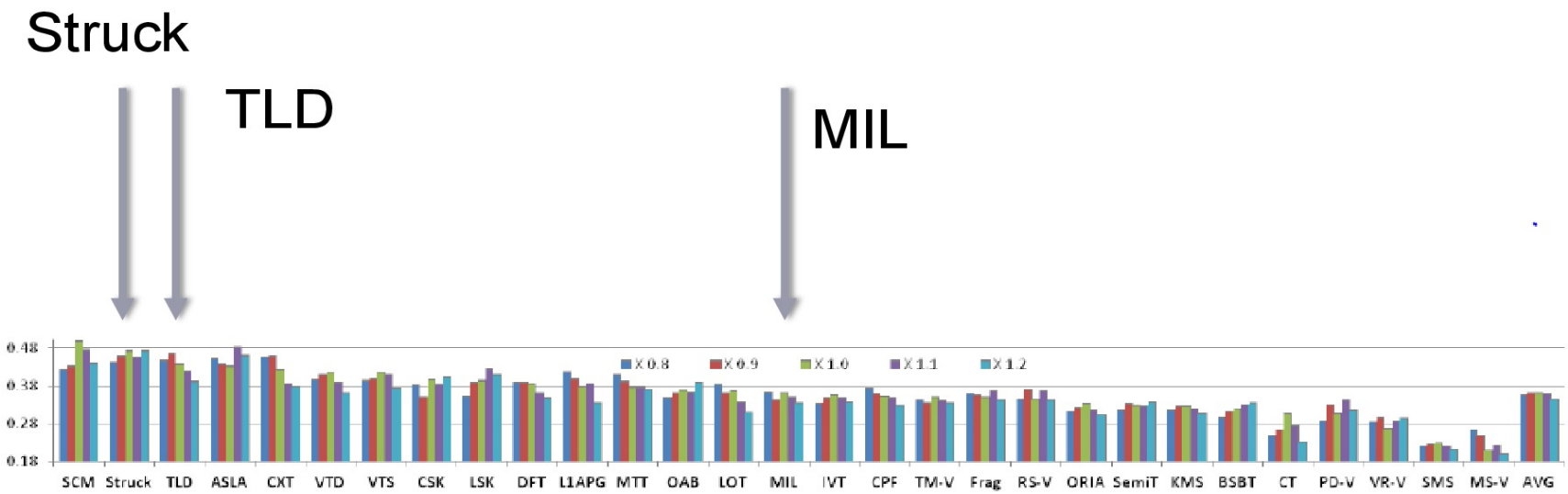


Figure 6. Performance summary for the trackers initialized with different size of bounding box. AVG (the last one) illustrates the average performance over all trackers for each scale.

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

Slide Credit: