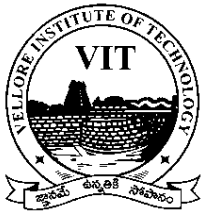


VIT-AP UNIVERSITY

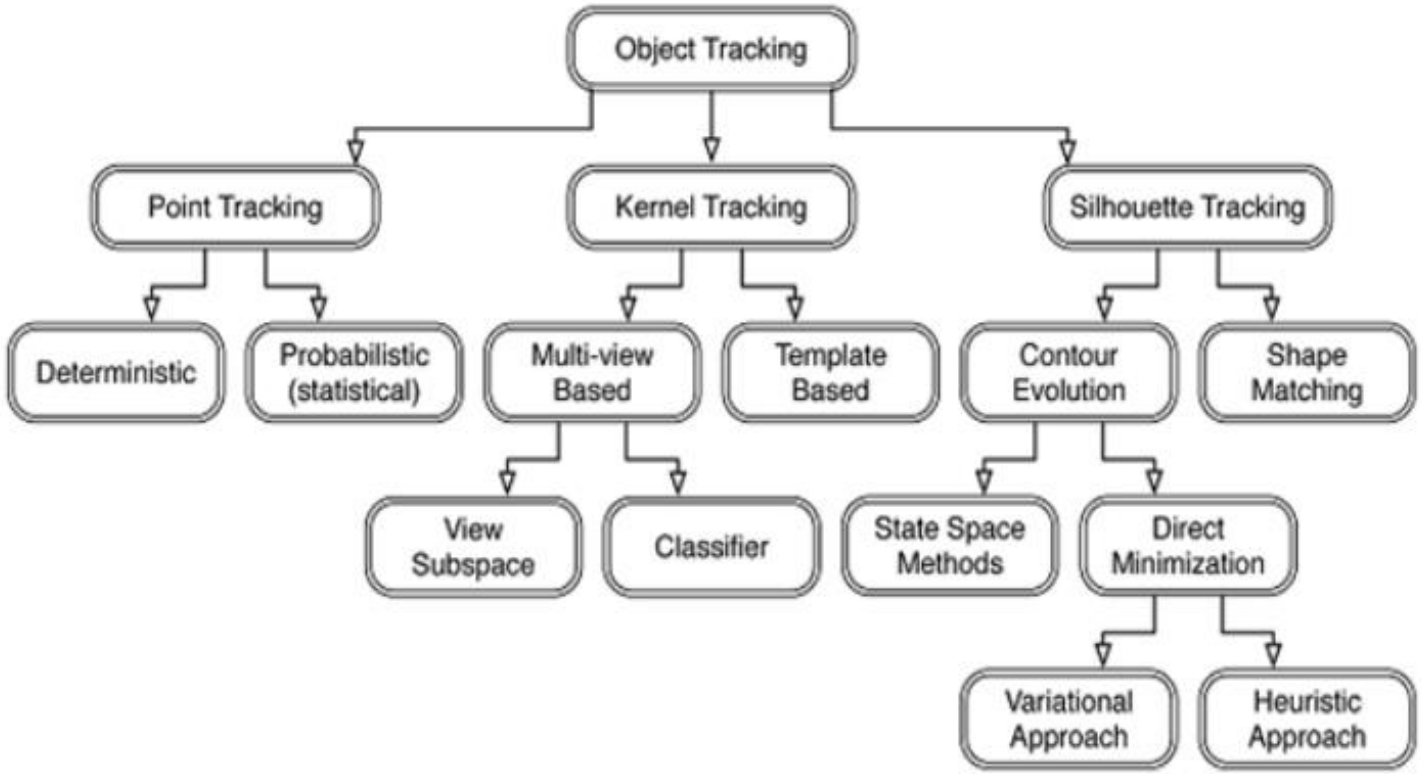
Computer Vision

(Course Code: 4047)

Module-4:Lecture-3: Object Tracking

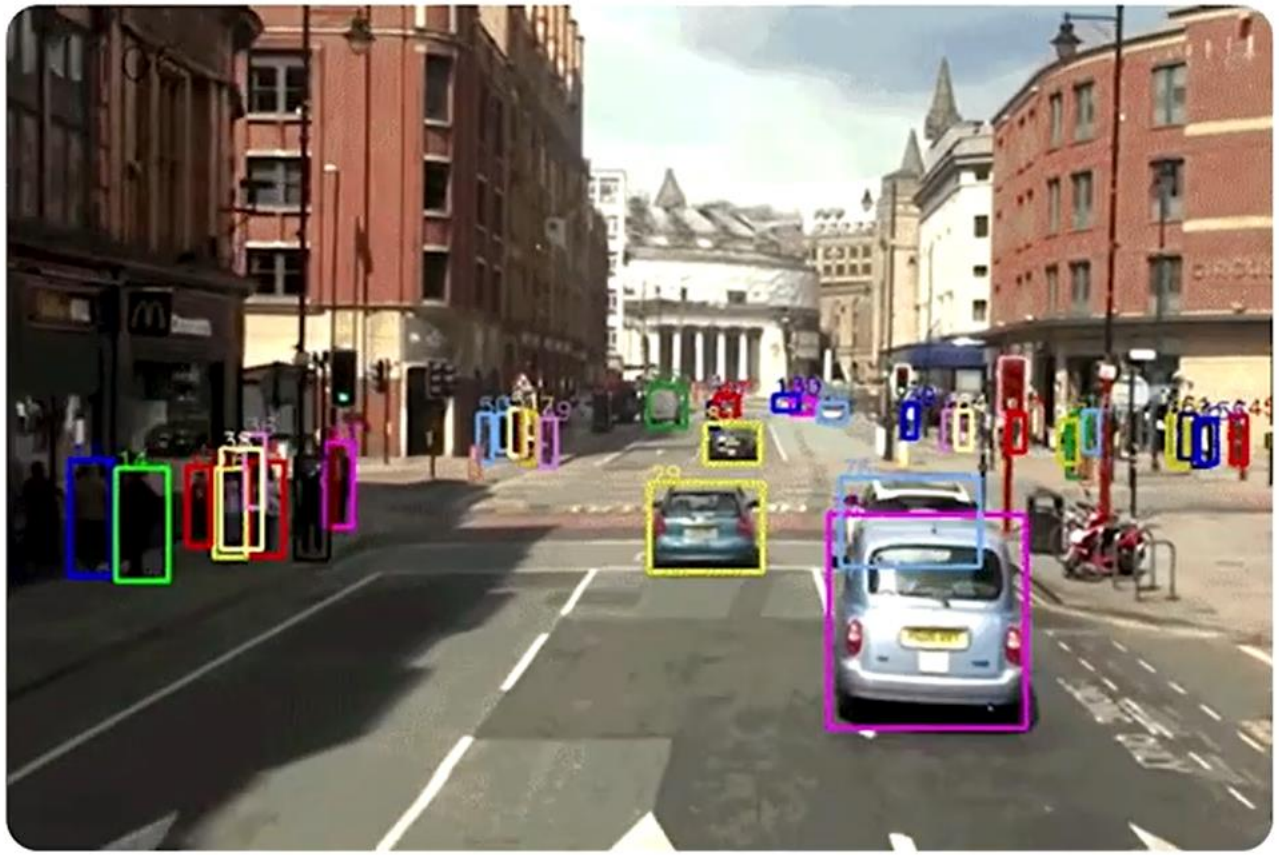
Gundimeda Venugopal, Professor of Practice, SCOPE

Object Tracking - Different Categories



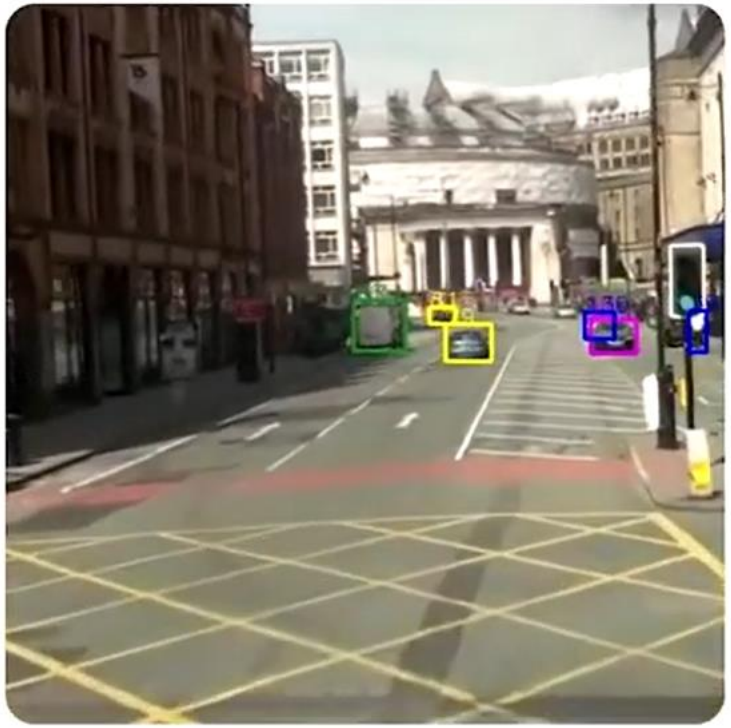
Multiple Object Tracking: Definition

Multiple Object Tracking (MOT) is the problem of identifying **multiple objects** in a video or live feed and representing them as a set of **trajectories**



Video: MOT Challenge

Challenges

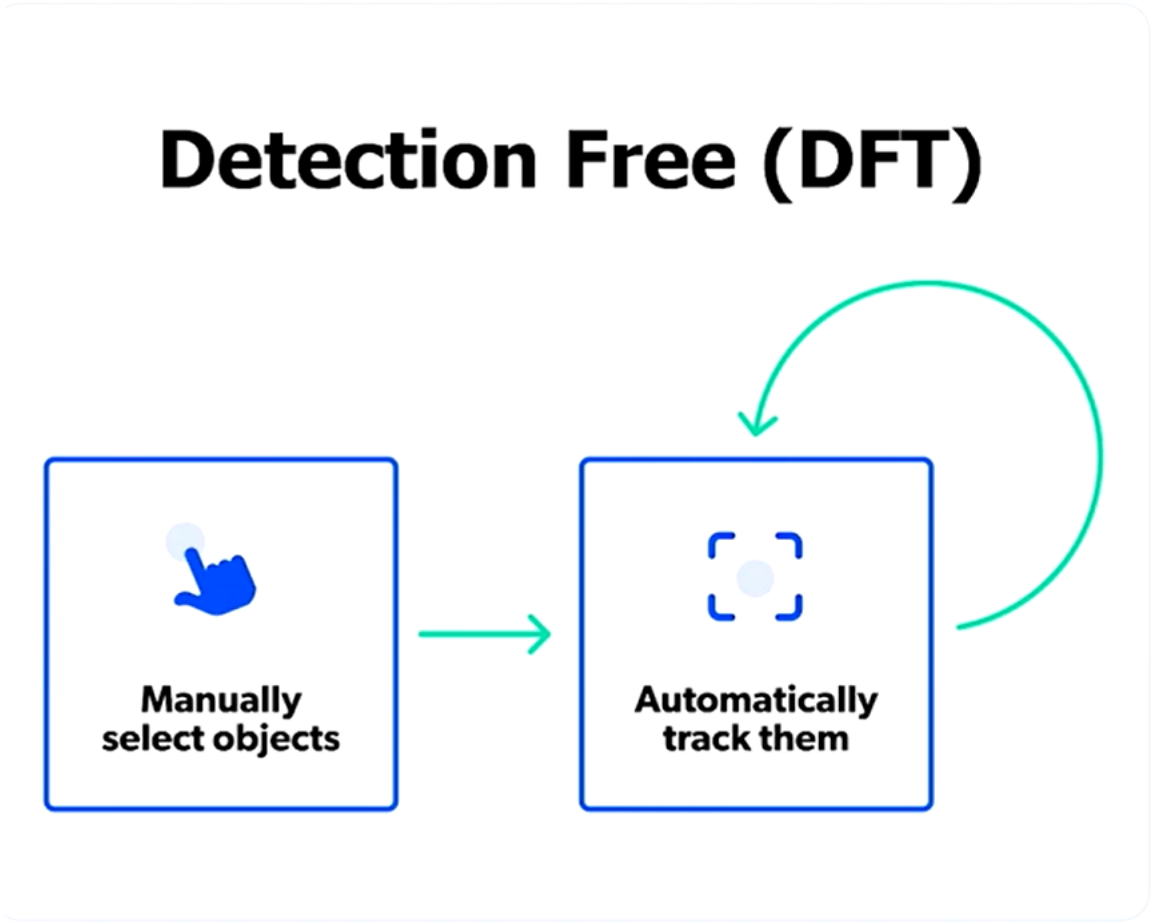


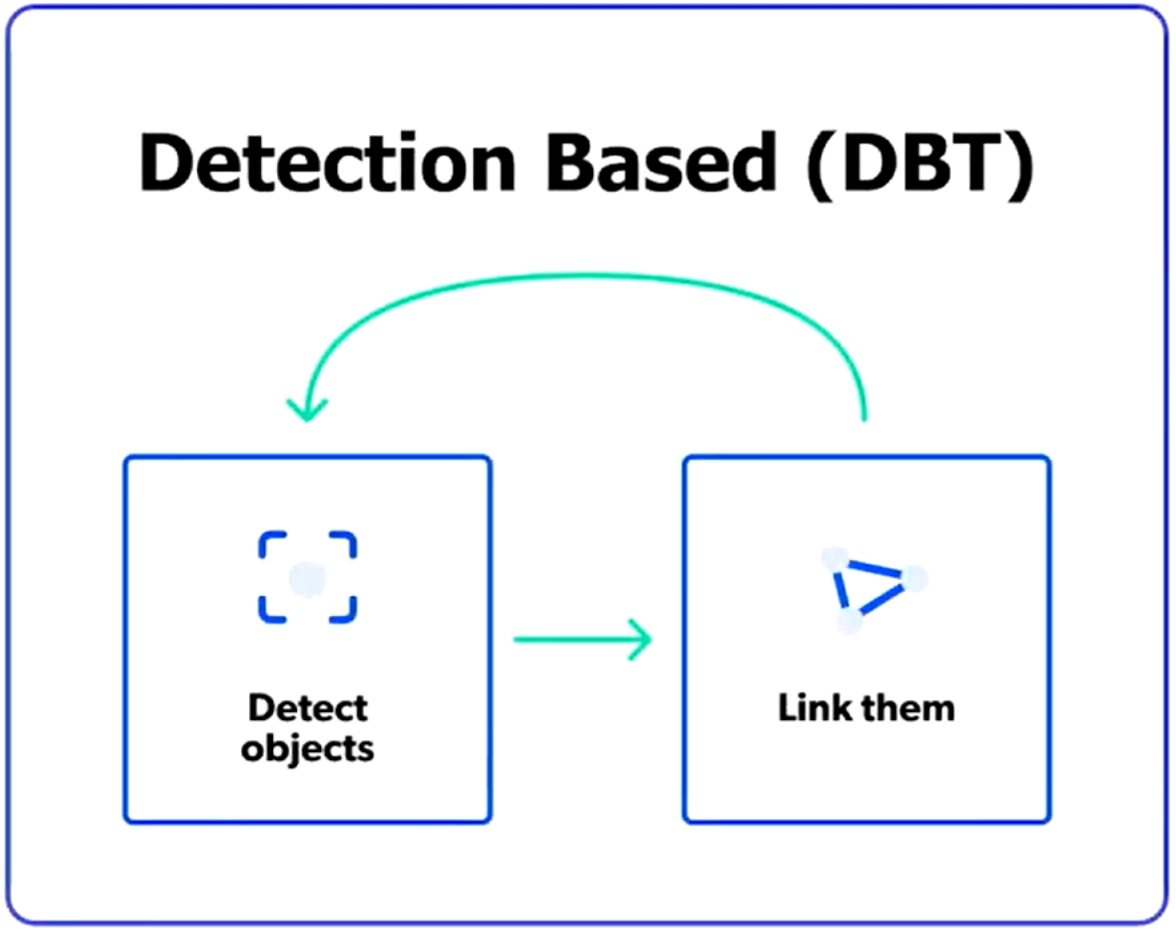
Changes in appearance

Occlusions

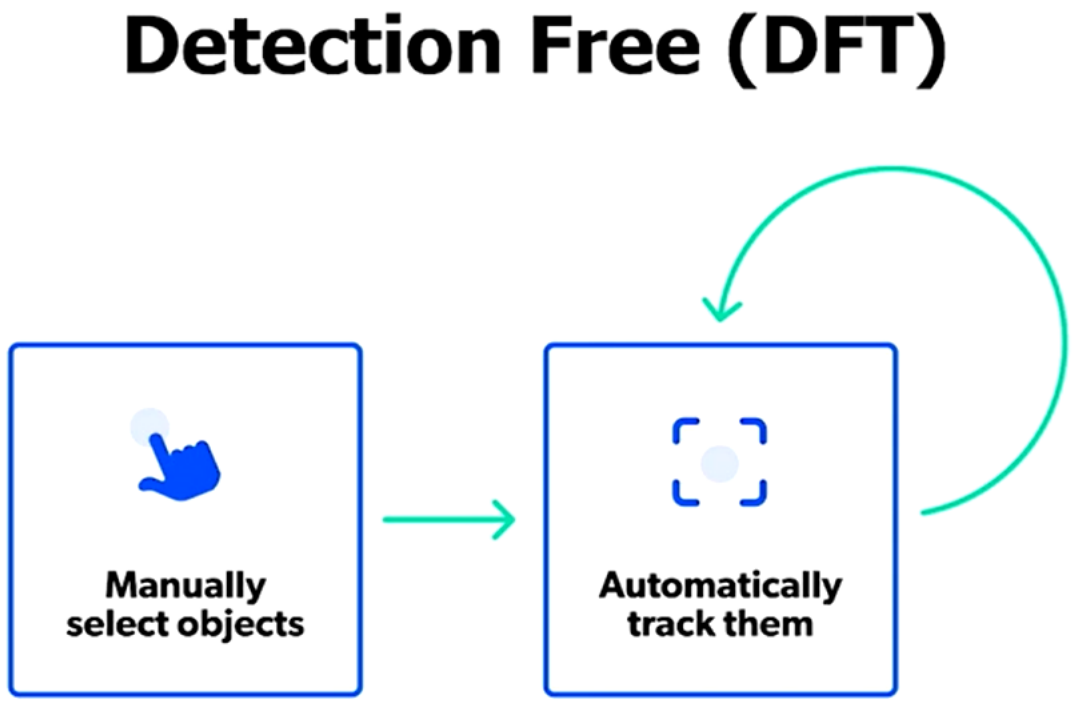
Crowded scenes

Building Blocks: Object Detection





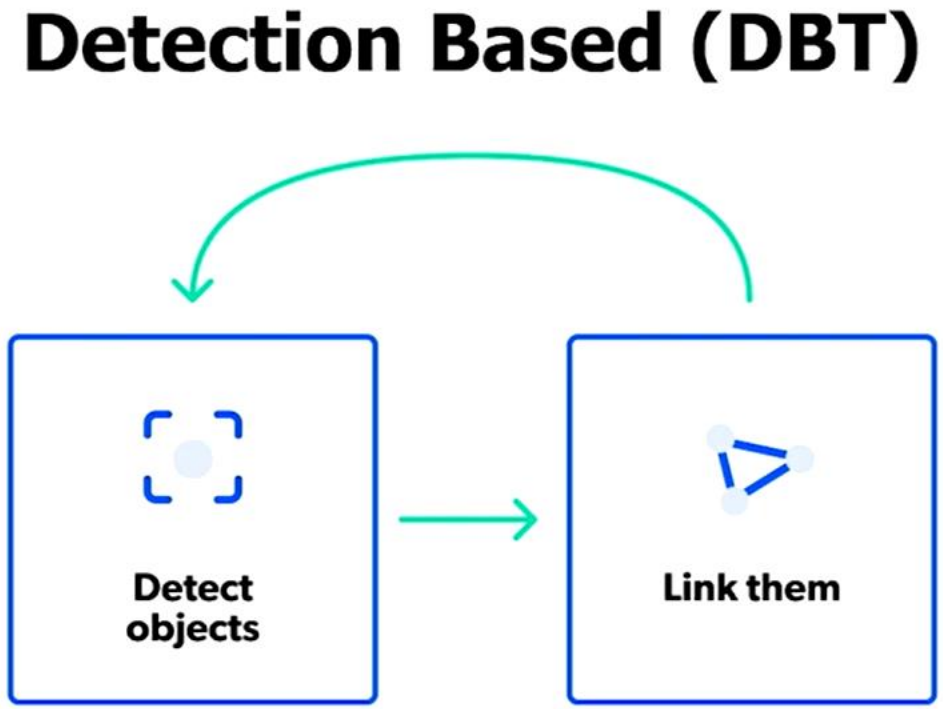
Detection Free Tracking



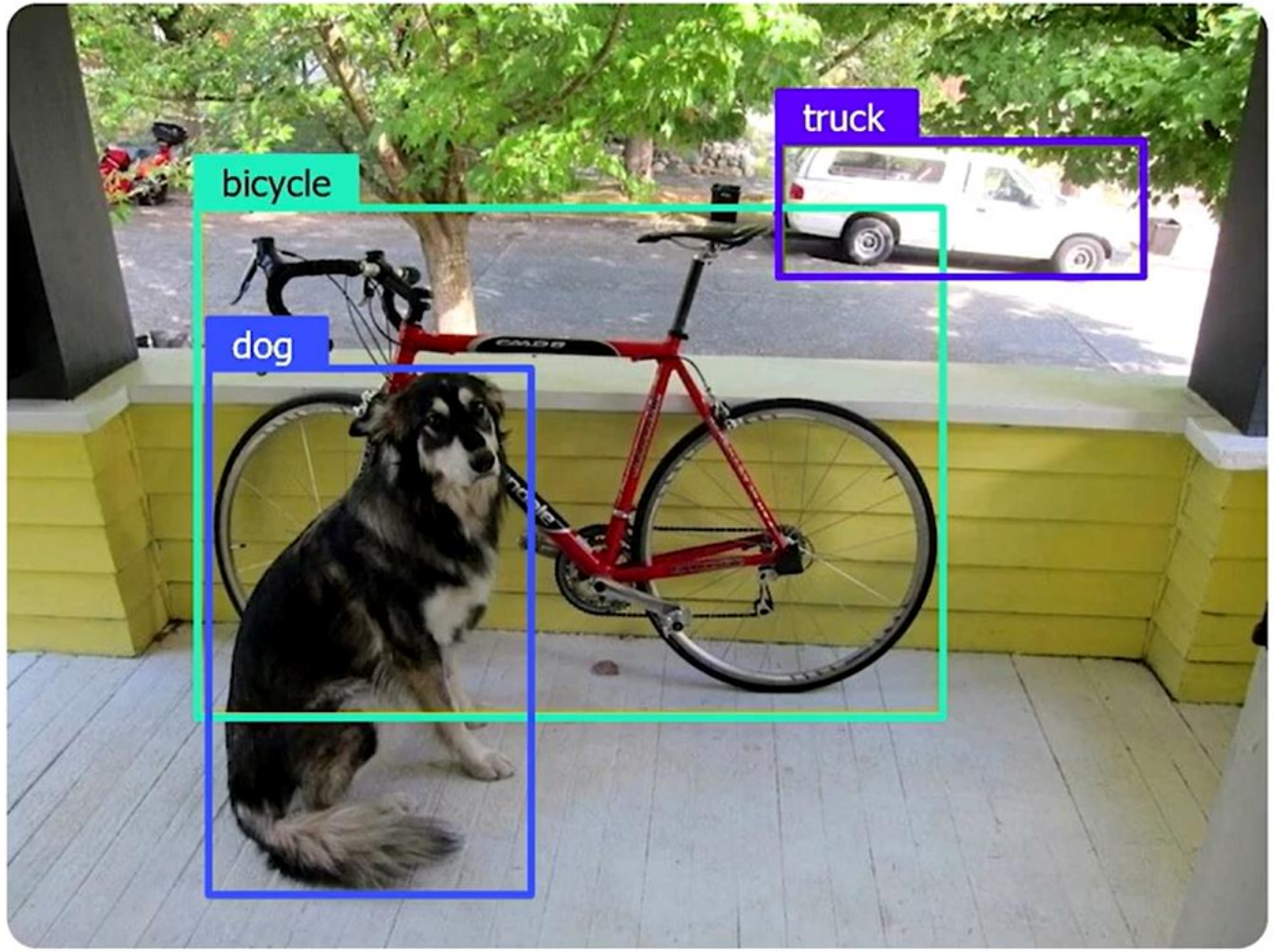
- Automatically track **manually** selected objects
- Can track **any** type of object
- **Can't** handle new objects coming into the scene

Detection based Tracking

- Detect objects in **each** frame
- New objects are **automatically** discovered
- Object classes need to be **predefined**



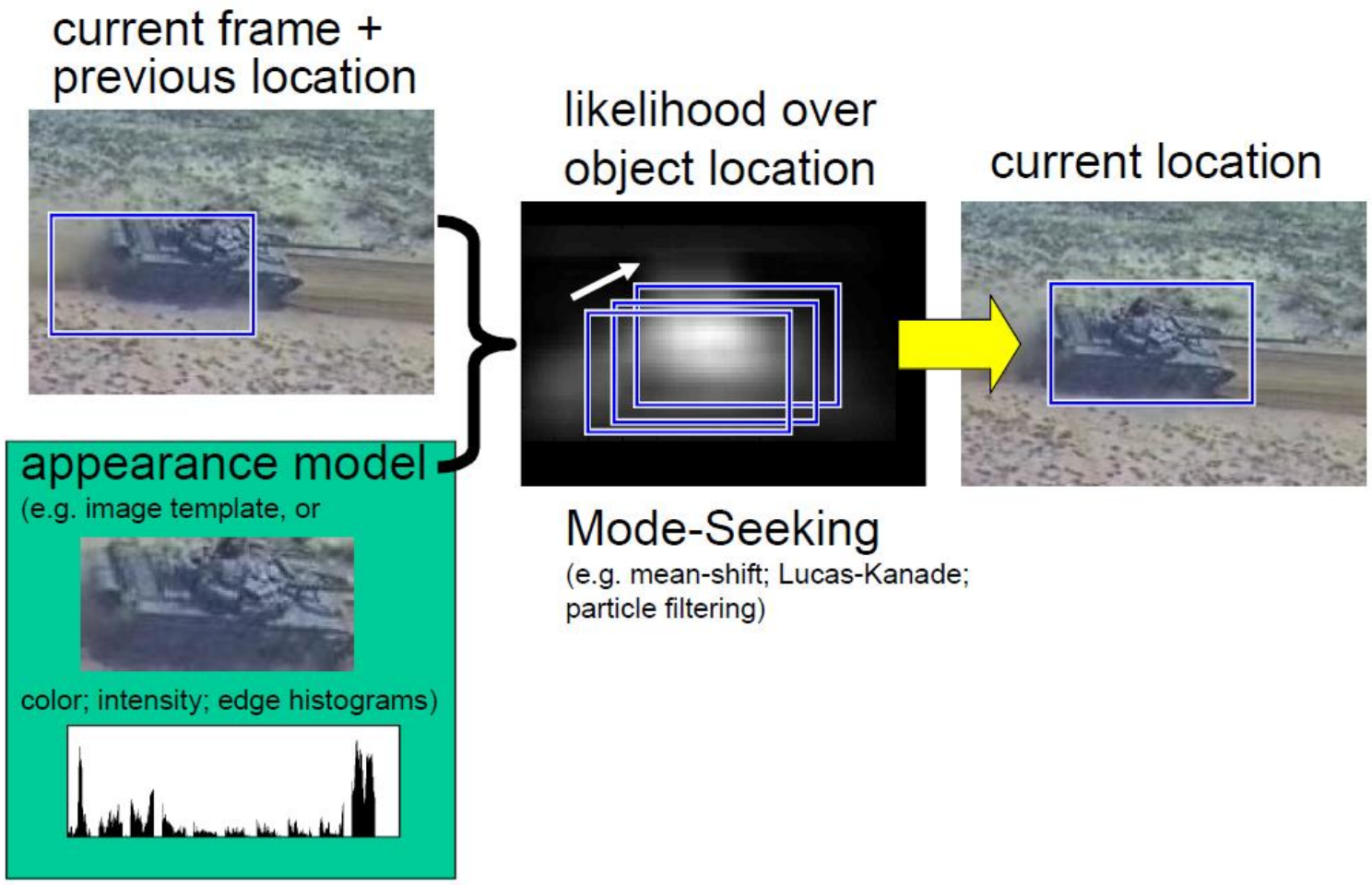
Object Tracking using a Deep learning model (e.g., YOLO)



Appearance based Tracking

- Appearance modeling is concerned with modeling an object's visual appearance.
- When the targeted object goes through numerous differing scenarios - such as different lighting conditions, angles or speeds - the appearance of the object may vary, resulting in misinformation and the algorithm losing track of the object.

- This approach consists primarily of two components:
  - **Visual representation**: constructing robust object descriptions using visual features
  - **Statistical modeling**: building effective mathematical models for object identification using statistical learning techniques

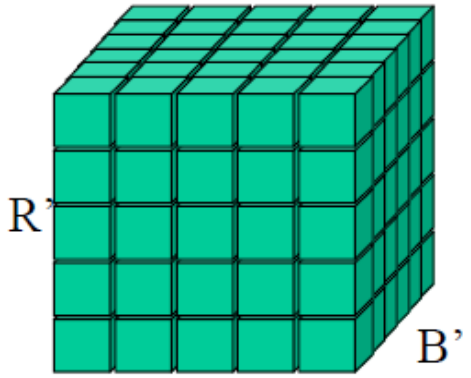


Histogram Appearance Models

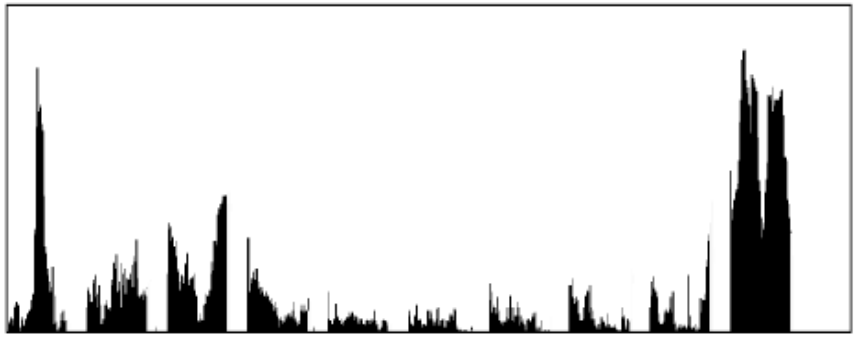
- Motivation – to track non-rigid objects, (like a walking person), it is hard to specify an explicit 2D parametric motion model.

- Appearances of non-rigid objects can sometimes be modeled with color distributions

Appearance via Color Histograms



discretize



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

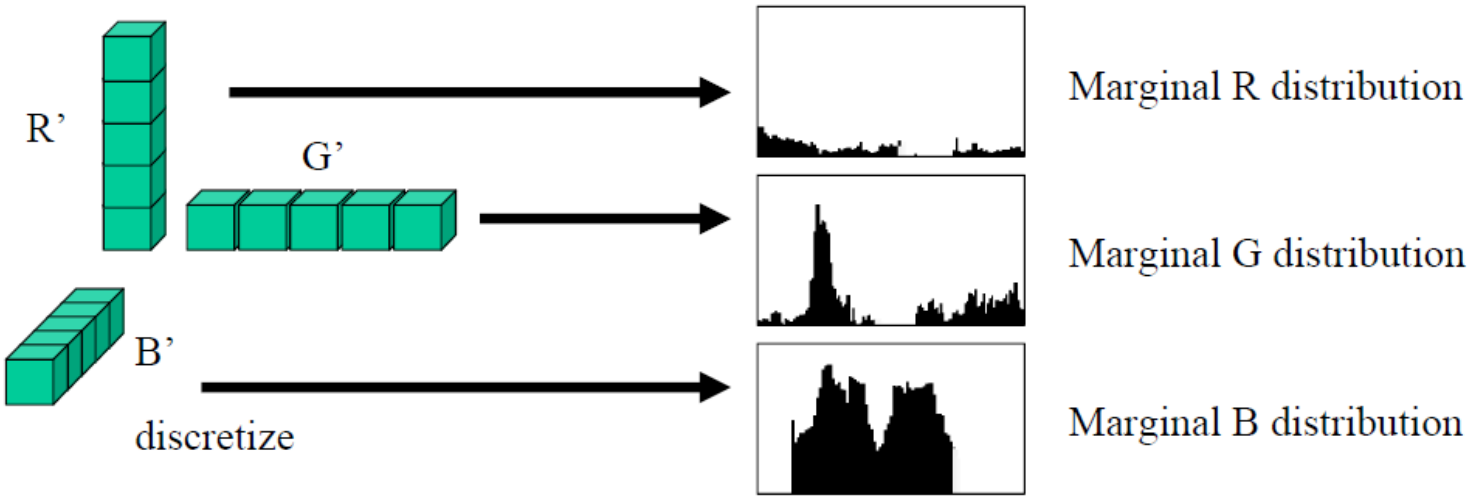
R' = R << (8 - nbits)
G' = G << (8 - nbits)
B' = B << (8-nbits)

Total histogram size is (2^(8-nbits))^3 (~16 million)

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

Smaller Color Histograms

Histogram information can be much much smaller if we are willing to accept a loss in color resolvability.

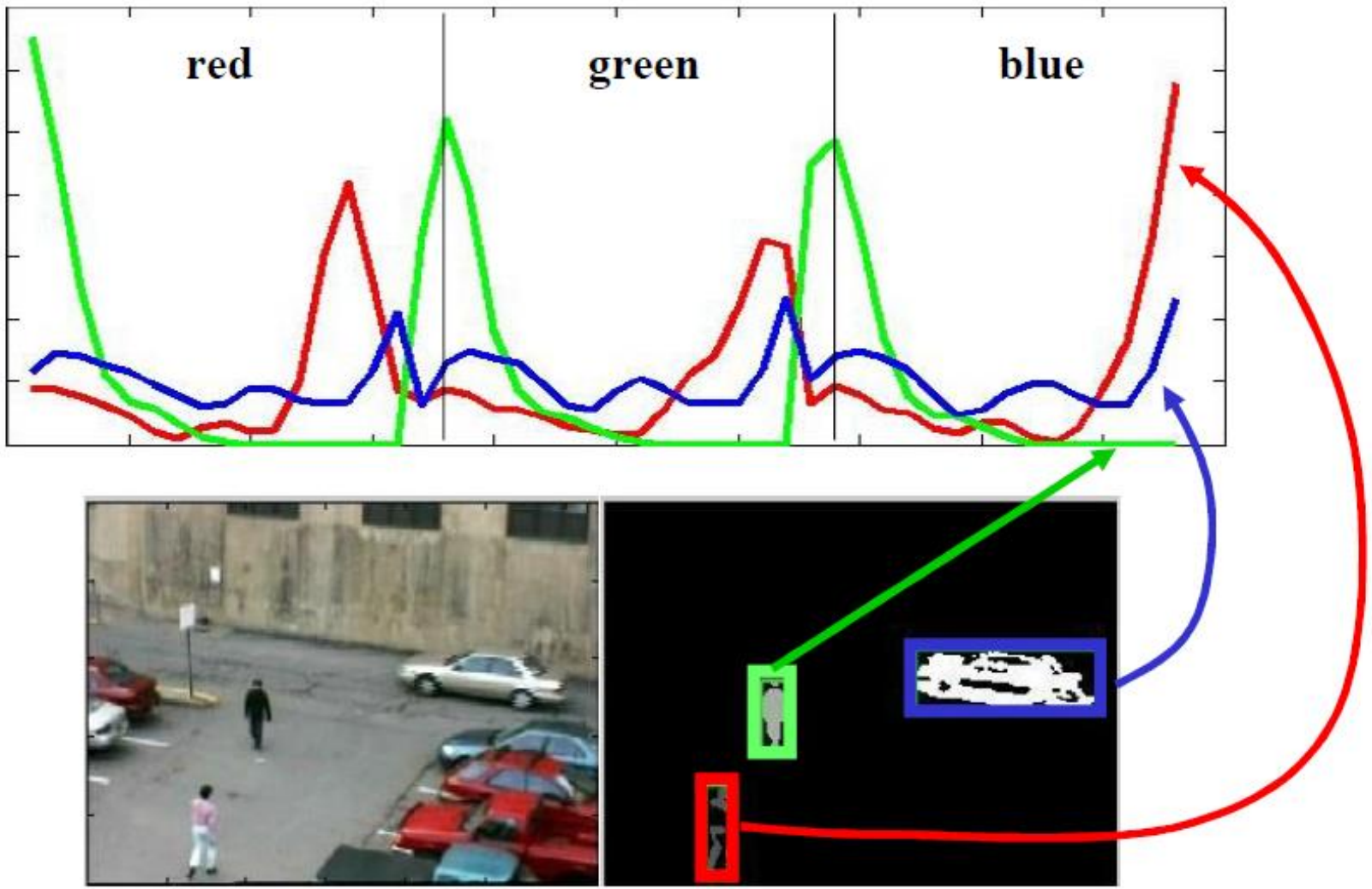


R' = R << (8 - nbits)
G' = G << (8 - nbits)
B' = B << (8-nbits)

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

example, 4-bit encoding of R,G and B channels yields a histogram of size 3*16 = 48.

Color Histogram Example

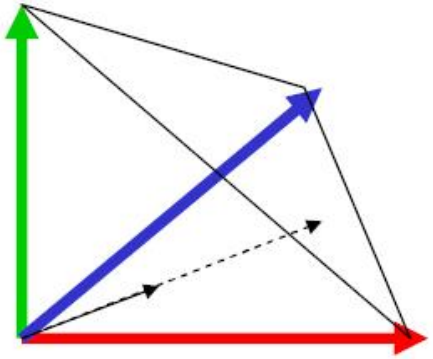


Normalized Color

(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.

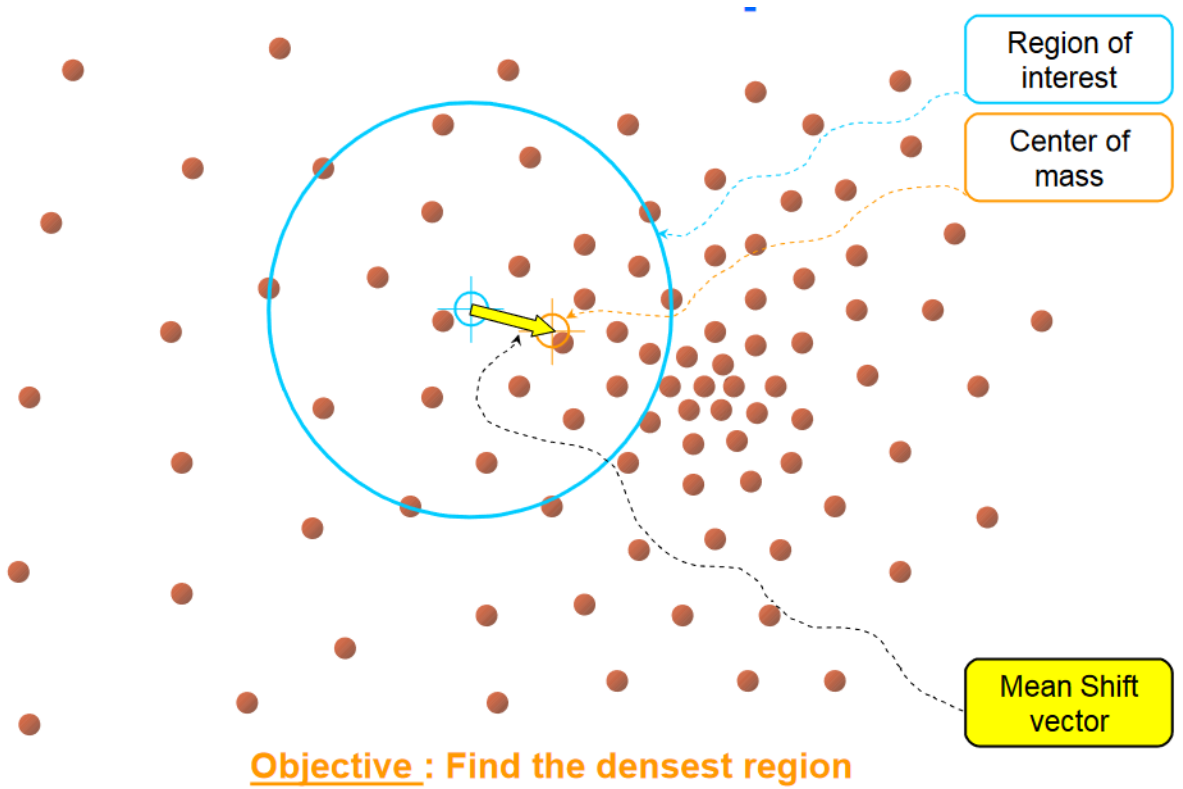


Mean Shift

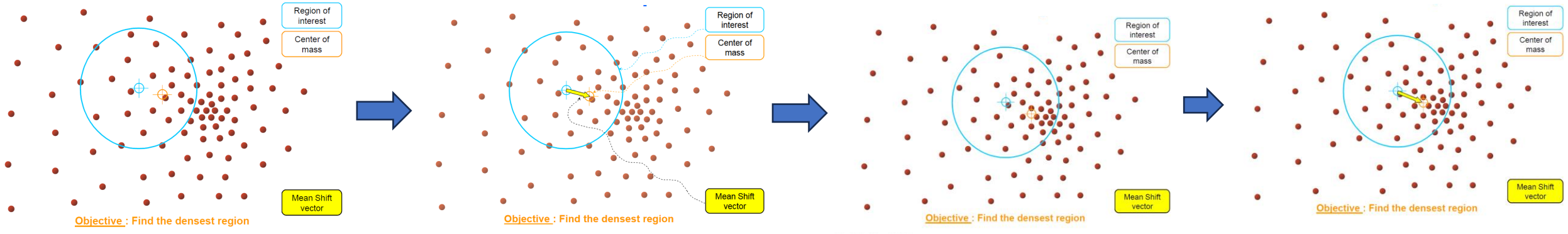
Mean Shift: Intuitive Description

Mean Shift is an algorithm that iteratively shifts a data point to the average of data points in its neighbourhood.

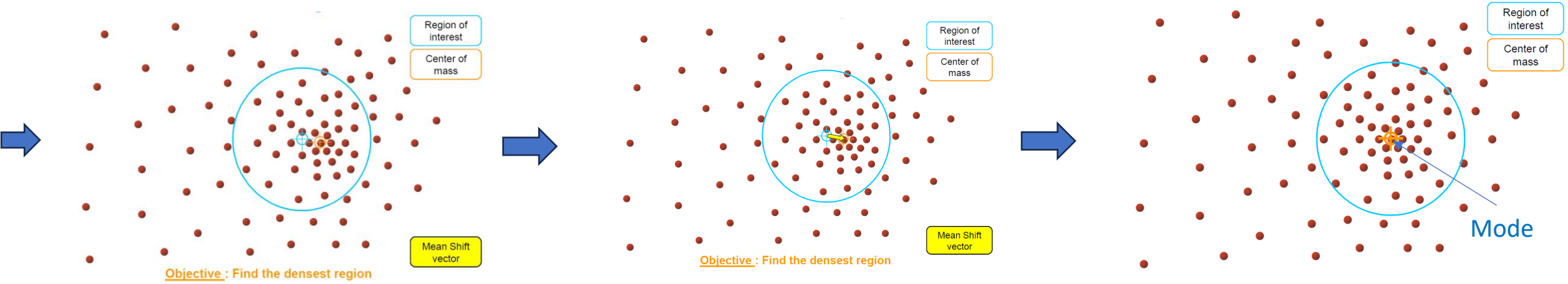
Mean Shift is useful for tracking, clustering, mode seeking and probability density estimation.



Mean Shift: Intuition



Every shift of data point is defined by a Mean-Shift vector which always points toward the maximum density increase.



Objective : Find the densest region

Mean Shift algorithm brief overview

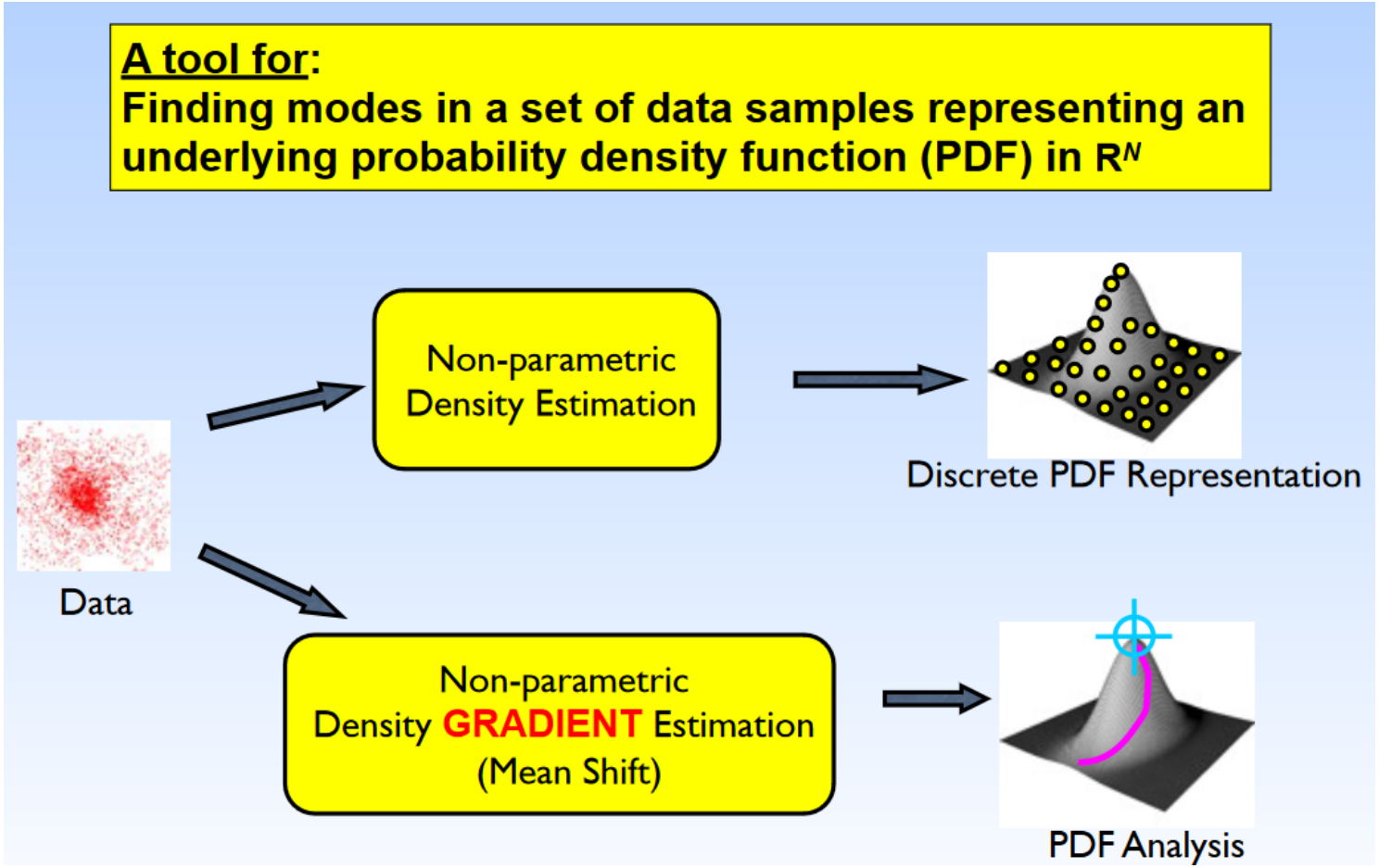
- The **Mean Shift algorithm** is based on several key steps. Firstly, the data must be formatted and pre-processed to select the appropriate features and avoid any bias in the results.
- Initial centroids are then selected from the **data itself or randomly.** These serve as starting points for iterations of the algorithm.
- Next comes the iteration stage, during which the points are moved according to the density of the **surrounding points**. The centroids are updated each time and the points are moved in the direction of the regions of maximum density.
- This iteration continues until the points converge towards the local modes and the centroids no longer move significantly.
- Different convergence criteria can be used, such as the distance between successive centroids or the decrease in total variation.
- One of the key parameters of this algorithm is the size of the search window. It influences the maximum distance over which points are moved, and can have a significant impact on the clustering results.
- The type of kernel must also be **selected according to the characteristics of the data and the desired results.** The most commonly used are Gaussian and uniform. This choice affects the way in which the density is estimated, and therefore the shape and size of the clusters obtained.

Non- Parametric Distribution

- Usually, to analyze the distribution of data points we can use different types of parametric distributions. e.g., Gaussian, Poisson, binomial,
- In Gaussian distribution, once we have samples we can compute the parameters (μ,σ) and have a simple description of a density function.
- The problem with this approach is that often we will not be able to fit the Gaussian distribution to our samples.
- If that is the case we need to use the non-parametric distribution.

What is Mean Shift?

Mean Shift is an algorithm that iteratively shifts a data point to the average of data points in its neighbourhood. The mean-shift algorithm is an efficient approach to tracking objects whose appearance is defined by color. (not limited to only color, however. Could also use edge orientations, texture, motion)



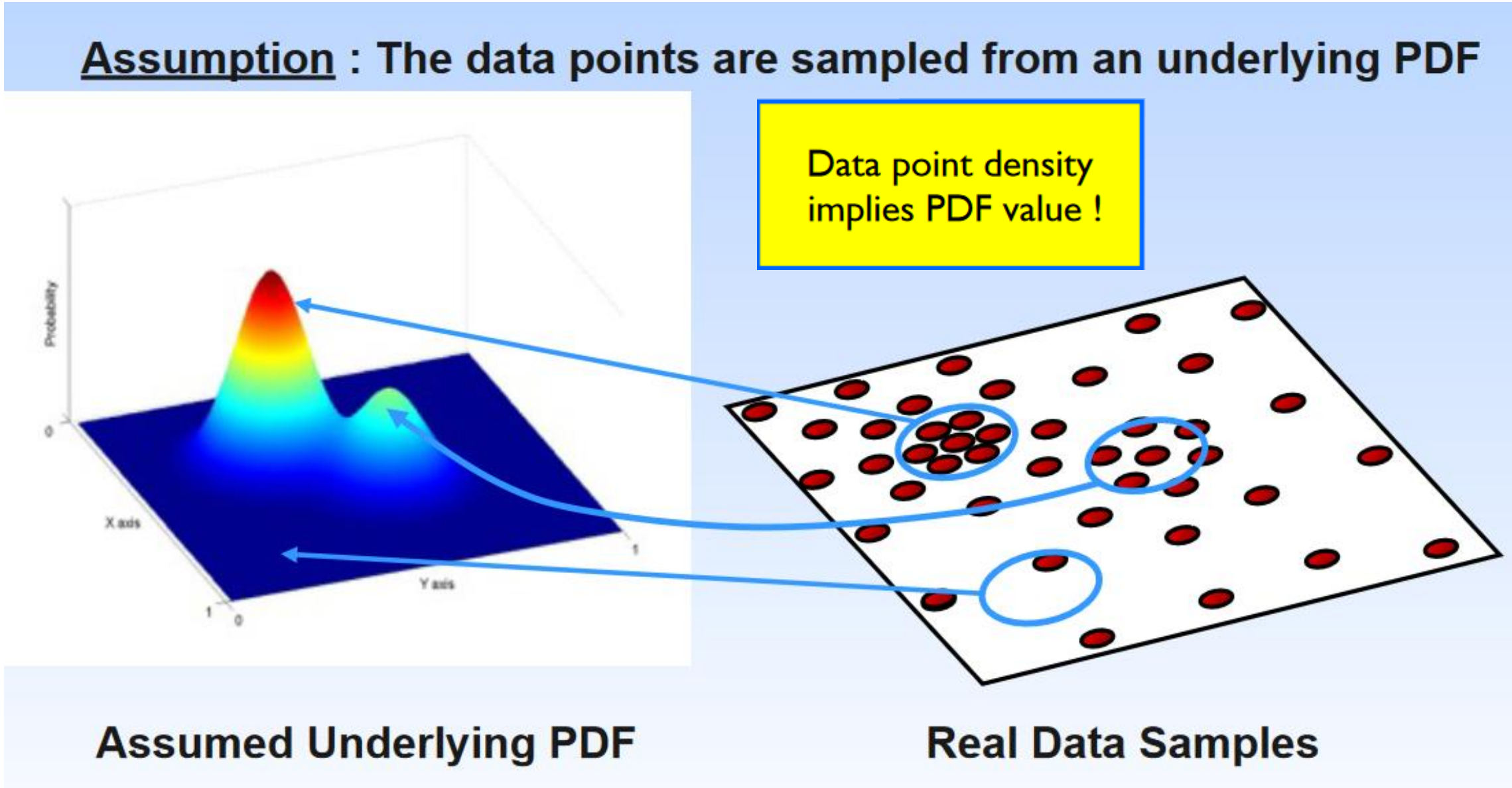
PDF in feature space
- Color space
- Scale space
- Actually any feature space you can conceive
- …

Modes are the local maxima of density in some feature space. You can say Modes are the peaked areas of a distribution or Probability Density.

There can be multiple modes (each associated with different cluster) in the sample space.

A set of samples drawn are from a Probability Distribution. But, does not belong to a standard distribution.

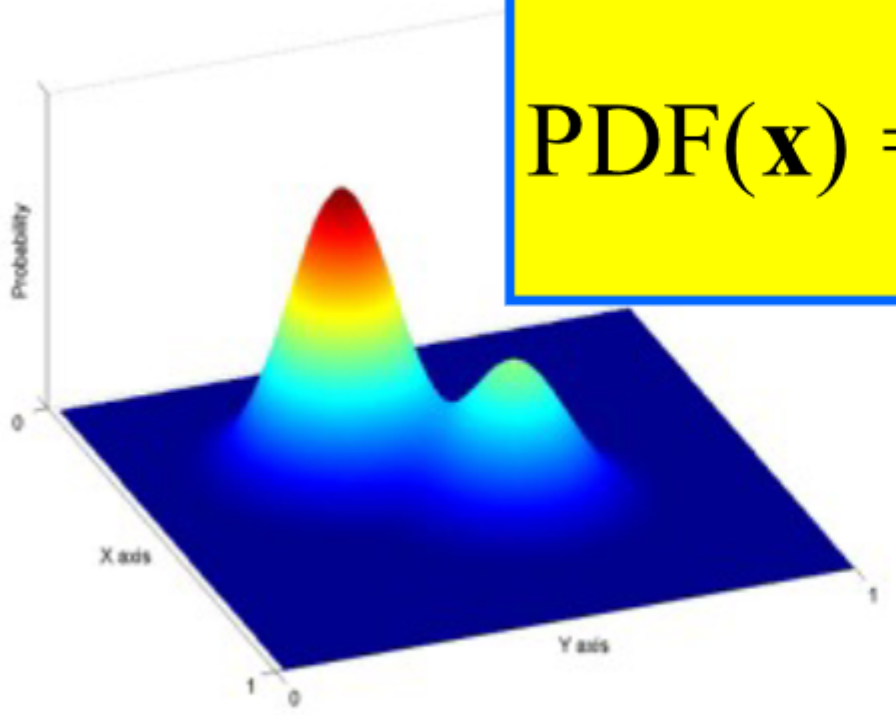
Non-Parametric Density Estimation

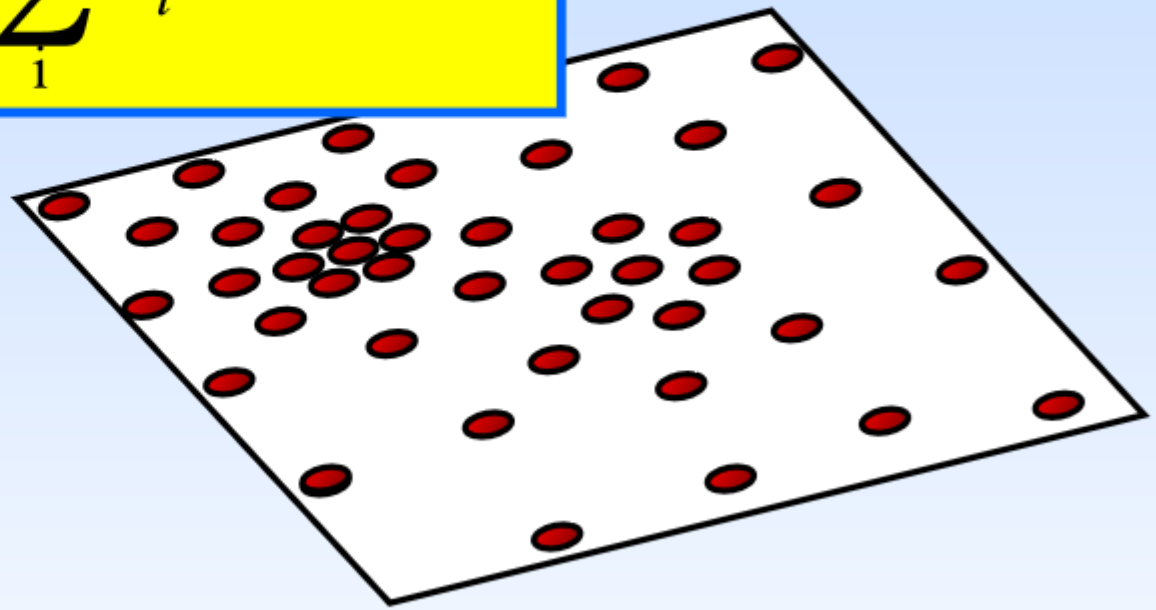


Parametric Density

Assumption : The data points are sampled from an underlying PDF

$$\mathrm{PDF}(\mathbf{x}) = \sum_{1} c_i \times e^{-\frac{(\mathbf{x}-\mu_i)^2}{2\sigma_i^2}}$$





Assumed Underlying PDF

Real Data Samples

Mean Shift Algorithm

- Consider a set S of n data points $\mathbf{x}_i$ in d-D Euclidean space X.
- Let $K(\mathbf{x})$ denote a kernel function that indicates how much $\mathbf{x}$ contributes to the estimation of the mean.
- Then, the sample mean $\mathbf{m}$ at $\mathbf{x}$ with kernel K is given by

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

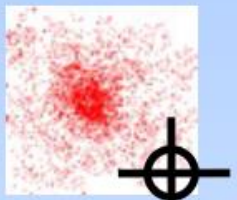
- The difference $\mathbf{m}(\mathbf{x}) - \mathbf{x}$ is called mean shift.
- Mean shift algorithm: iteratively move date point to its mean.
- In each iteration, $\mathbf{x} \leftarrow \mathbf{m}(\mathbf{x})$.
- The algorithm stops when $\mathbf{m}(\mathbf{x}) = \mathbf{x}$.
- The sequence $\mathbf{x}, \mathbf{m}(\mathbf{x}), \mathbf{m}(\mathbf{m}(\mathbf{x})), \ldots$ is called the trajectory of $\mathbf{x}$.
- If sample means are computed at multiple points, then at each iteration, update is done simultaneously to all these points.

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 $x_1 \ldots x_n$



Data

Examples:

- Epanechnikov Kernel

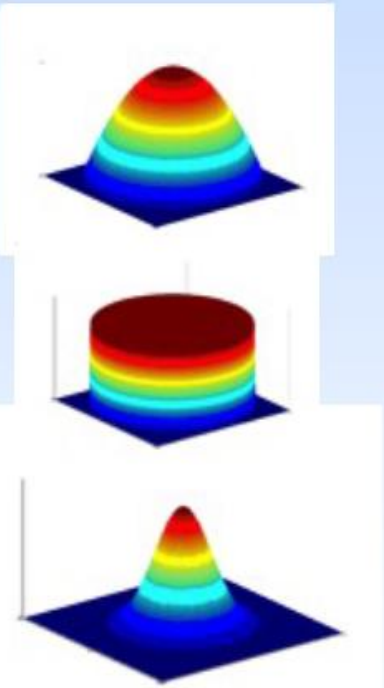
$$K_E(\mathbf{x}) = \begin{cases} c\left(1 - \|\mathbf{x}\|^2\right) & \|\mathbf{x}\| \le 1 \\ 0 & \text{otherwise} \end{cases}$$

- Uniform Kernel

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

- Normal Kernel

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



Kernel Density (Gradient) Estimation

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

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

Give up estimating the PDF !
Estimate **ONLY** the gradient

Using the Kernel form:

$$K(\mathbf{x}-\mathbf{x}_i) = ck\left(\left\|\frac{\mathbf{x}-\mathbf{x}_i}{h}\right\|^2\right)$$

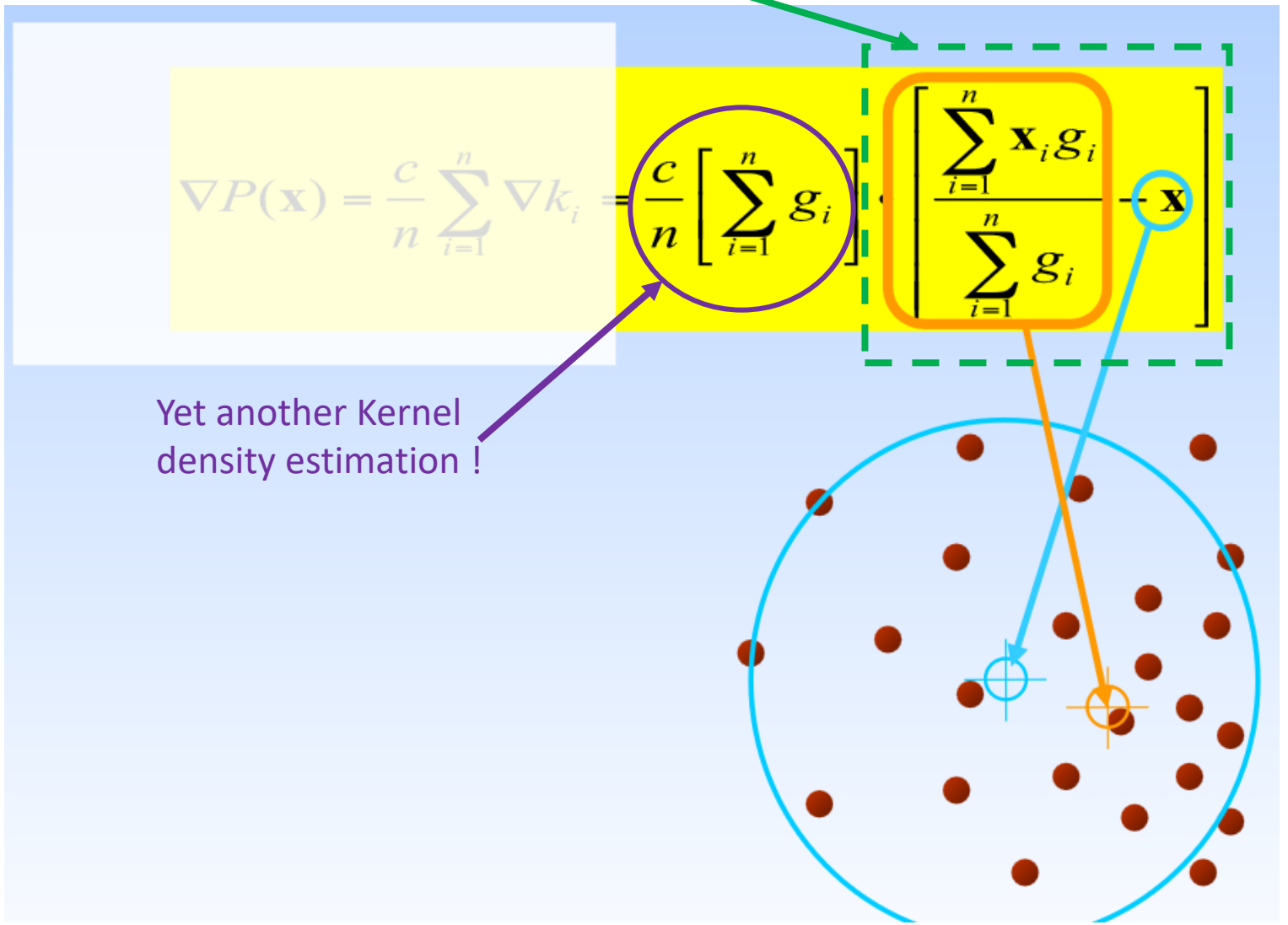
Size of window

We get :

$$\nabla P(\mathbf{x}) = \frac{c}{n}\sum_{i=1}^{n} \nabla k_i = \frac{c}{n}\left[\sum_{i=1}^{n} g_i\right] \cdot \left[\frac{\sum_{i=1}^{n} \mathbf{x}_i g_i}{\sum_{i=1}^{n} g_i} - \mathbf{x}\right]$$

$$g(\mathbf{x}) = -k'(\mathbf{x})$$

Computing the Mean Shift



Computing the Mean Shift

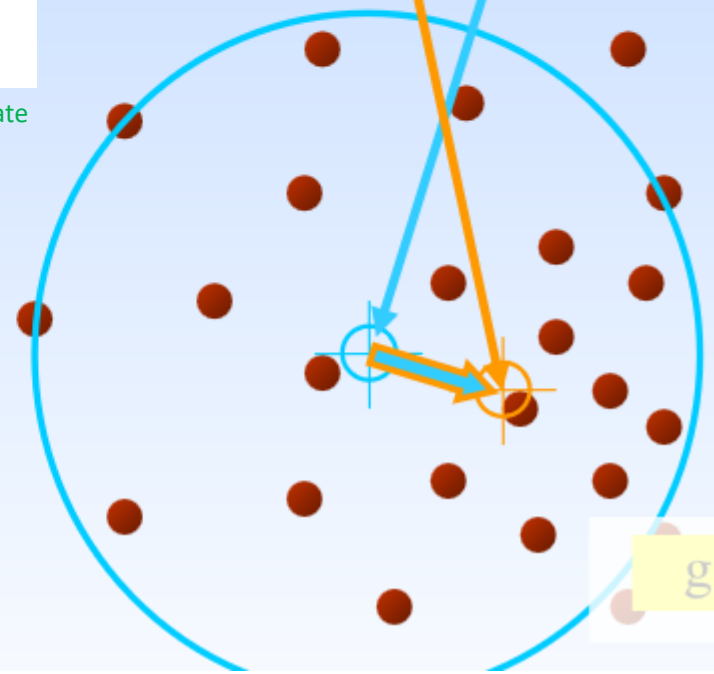
$$\nabla P(\mathbf{x}) = \frac{c}{n}\sum_{i=1}^{n}\nabla k_i = \frac{c}{n}\left[\sum_{i=1}^{n} g_i\right]\left[\frac{\sum_{i=1}^{n}\mathbf{x}_i g_i}{\sum_{i=1}^{n} g_i} - \mathbf{x}\right]$$

Gradient of the PDF is related to the mean shift vector

$$\nabla P(\boldsymbol{x}) \propto m(\boldsymbol{x})$$

The mean shift is a 'step' in the direction of the gradient of the KDE

Kernel density Estimate



$$g(\mathbf{x}) = -k'(\mathbf{x})$$

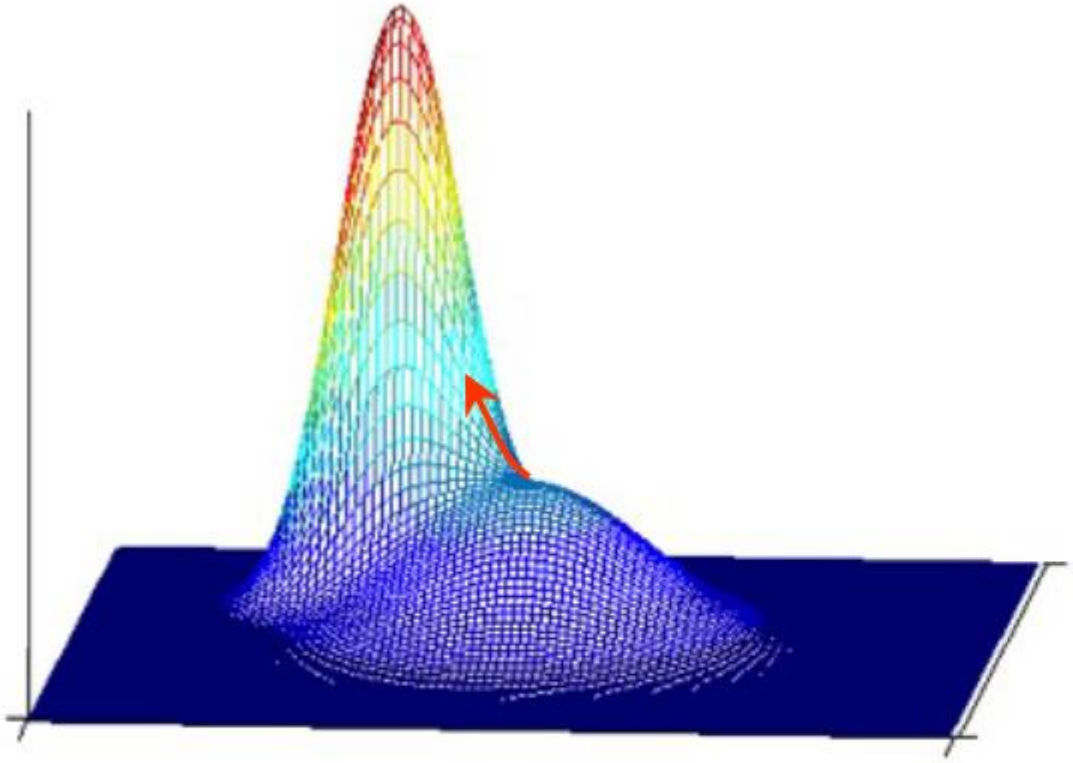
Simple Mean Shift procedure:

- Compute mean shift vector

$$\mathbf{m}(\mathbf{x}) = \left[\frac{\sum_{i=1}^{n}\mathbf{x}_i g\left(\frac{\|\mathbf{x}-\mathbf{x}_i\|^2}{h}\right)}{\sum_{i=1}^{n} g\left(\frac{\|\mathbf{x}-\mathbf{x}_i\|^2}{h}\right)} - \mathbf{x}\right]$$

- Translate the Kernel window by **m(x)**

Mean Shift Mode Detection



What happens if we reach a saddle point ?

Perturb the mode position and check if we return back

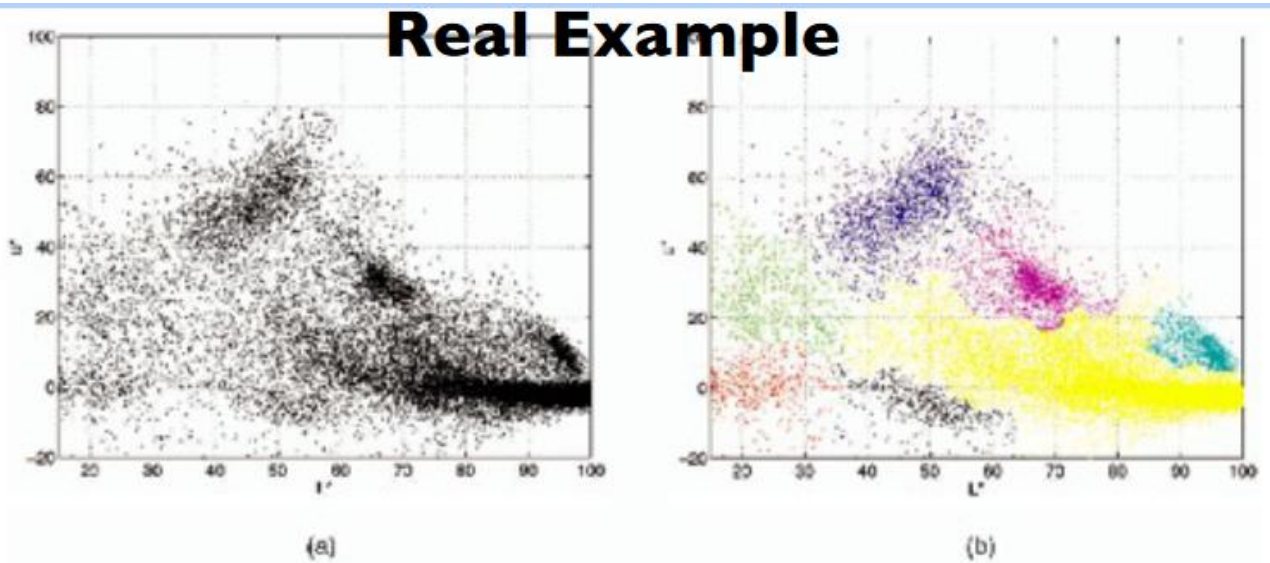
Updated Mean Shift Procedure:

- Find all modes using the Simple Mean Shift Procedure
- Prune modes by perturbing them (find saddle points and plateaus)
- Prune nearby – take highest mode in the window

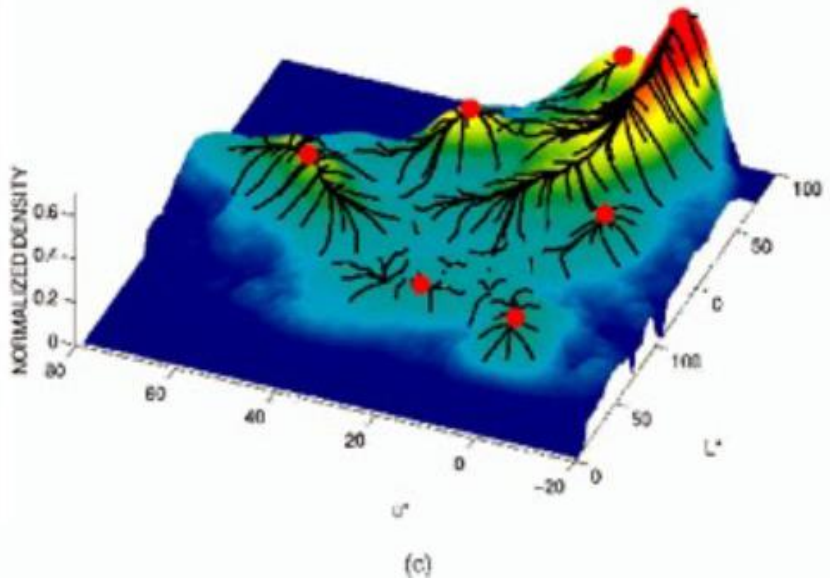
Clustering Example (Multiple Modes)

2D (L*u) space representation

Real Example



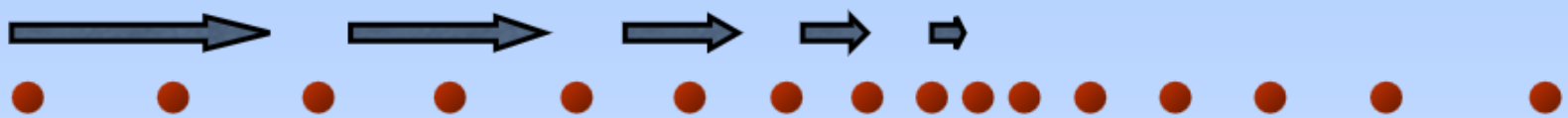
Final clusters



Not all trajectories in the attraction basin reach the same mode

Region surrounding each mode is part of a separate cluster

Mean Shift Properties

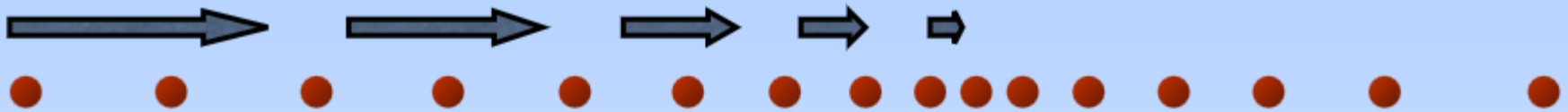


- Automatic convergence speed – the mean shift vector size depends on the gradient itself.
- Near maxima, the steps are small and refined

Adaptive Gradient Ascent

- Convergence is guaranteed for infinitesimal steps only → infinitely convergent, (therefore set a lower bound)
- For Uniform Kernel (), convergence is achieved in a finite number of steps
- Normal Kernel () exhibits a smooth trajectory, but is slower than Uniform Kernel ().

Mean Shift Strengths & Weaknesses



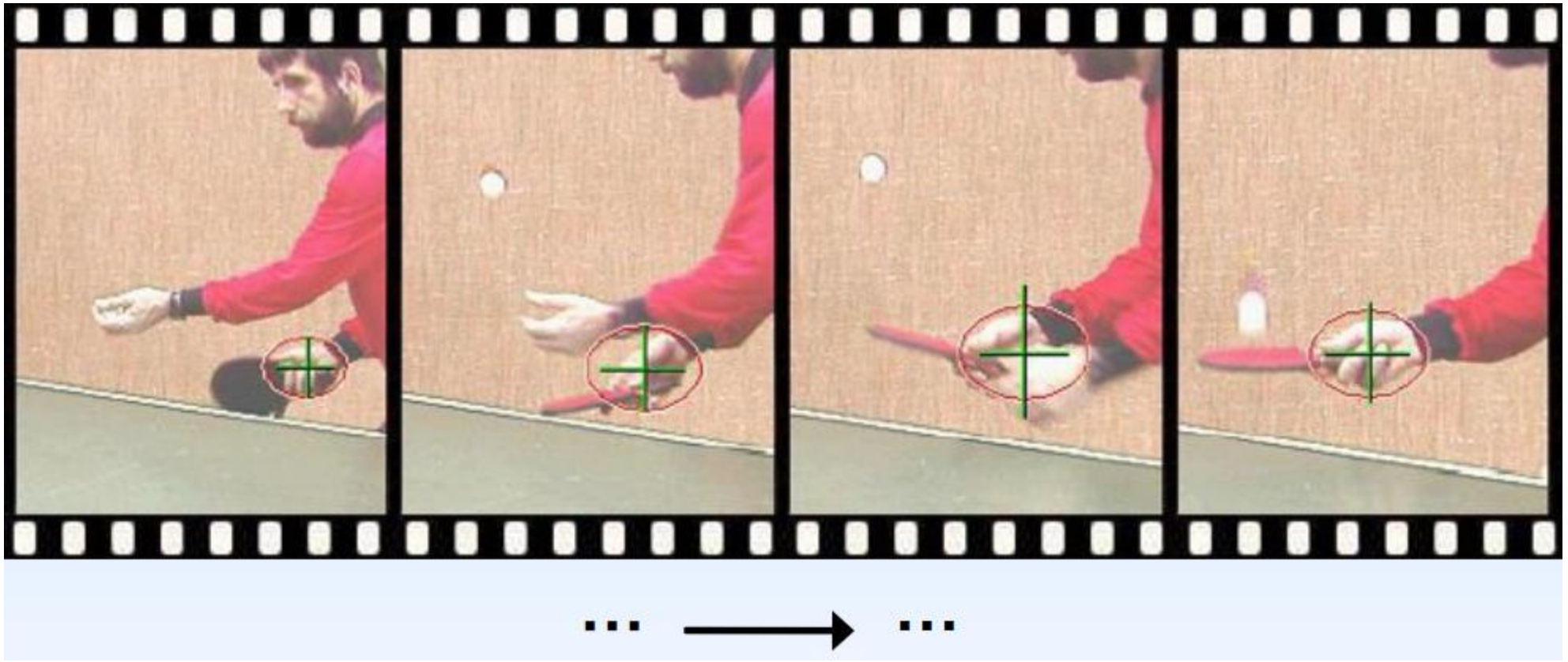
Strengths :

- Application independent tool
- Suitable for real data analysis
- Does not assume any prior shape (e.g. elliptical) on data clusters
- Can handle arbitrary feature spaces
- Only ONE parameter to choose
- *h* (window size) has a physical meaning, unlike K-Means

Weaknesses :

- The window size (bandwidth selection) is not trivial
- Inappropriate window size can cause modes to be merged, or generate additional "shallow" modes
- Use adaptive window size

Non-Rigid Object Tracking



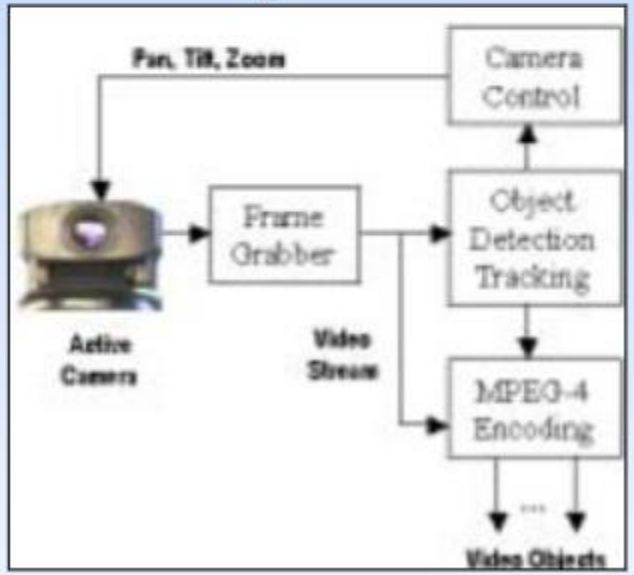
Non-Rigid Object Tracking

Real-Time

Surveillance

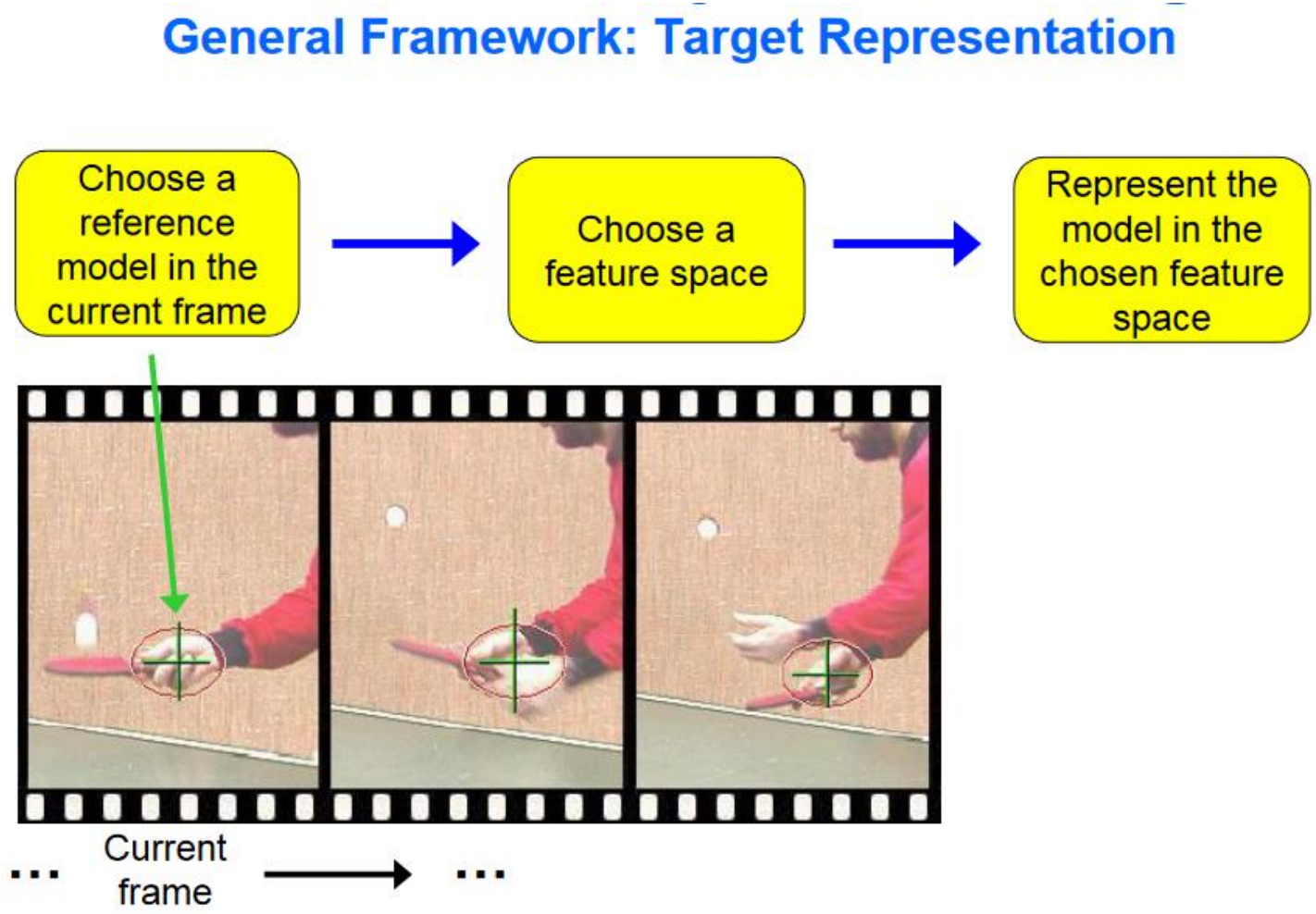
Driver Assistance

Object-Based Video Compression

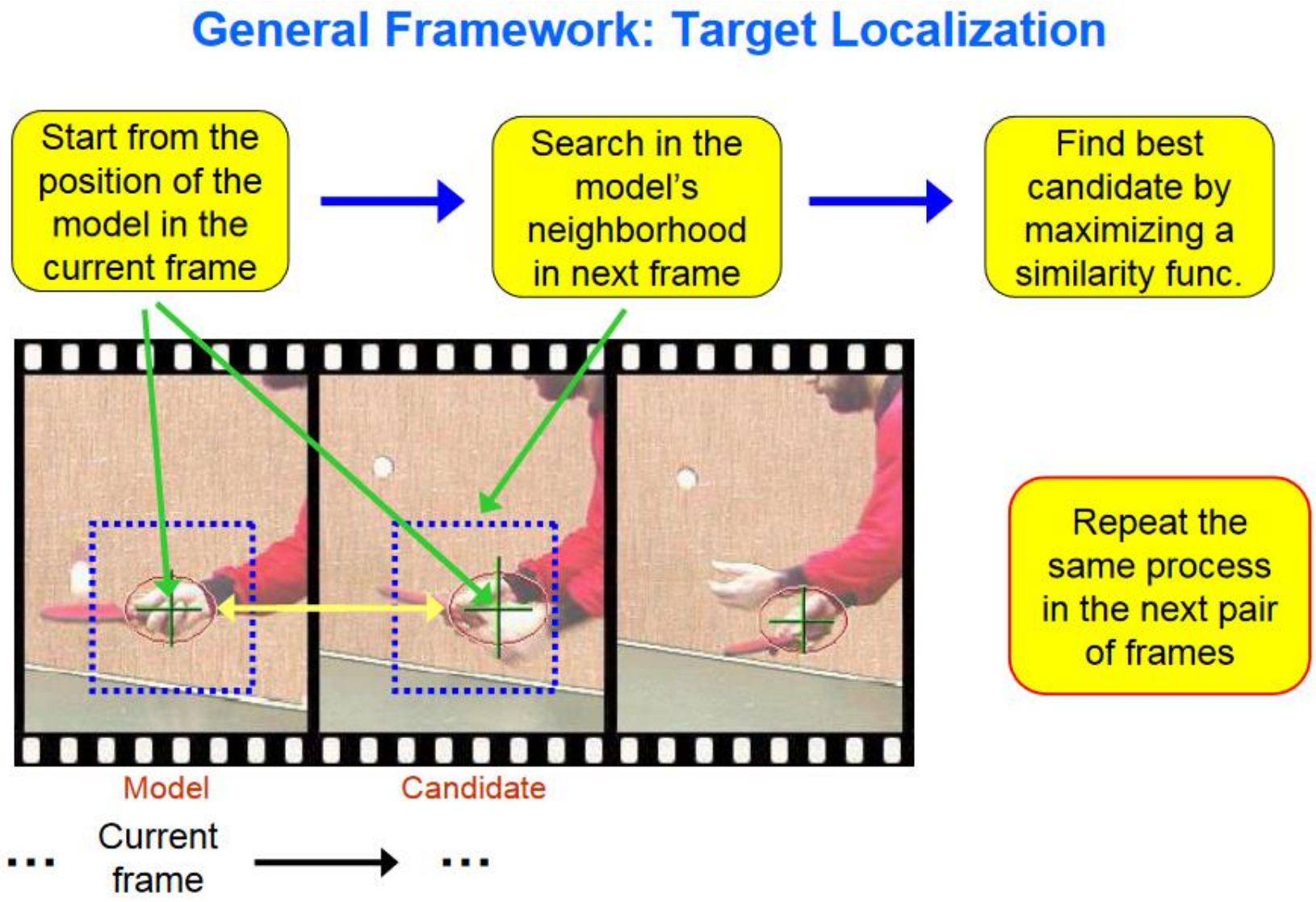


Mean Shift Object Tracking

Mean-Shift Object Tracking



Mean-Shift Object Tracking



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.

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

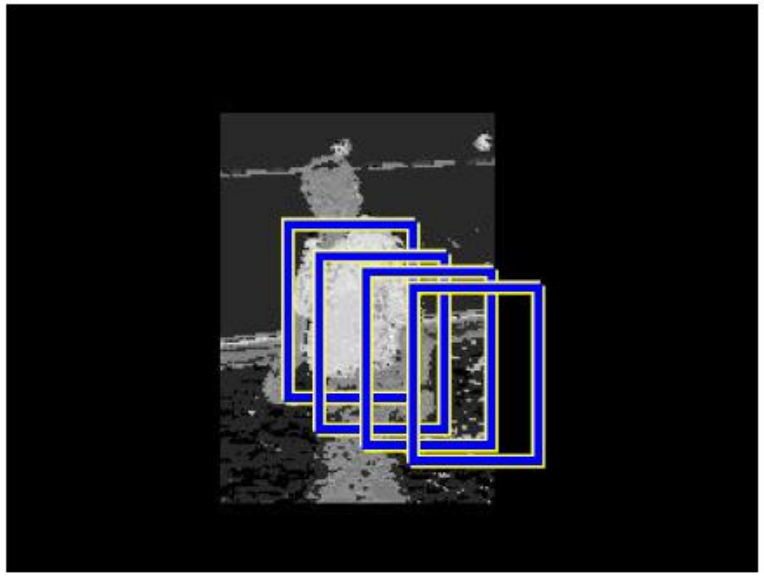
- color
- texture
- shape (boundary)
- predicted location

Note: So far, we have described mean-shift as operating over a set of point samples...

Mean shift example

Mean-Shift Tracking

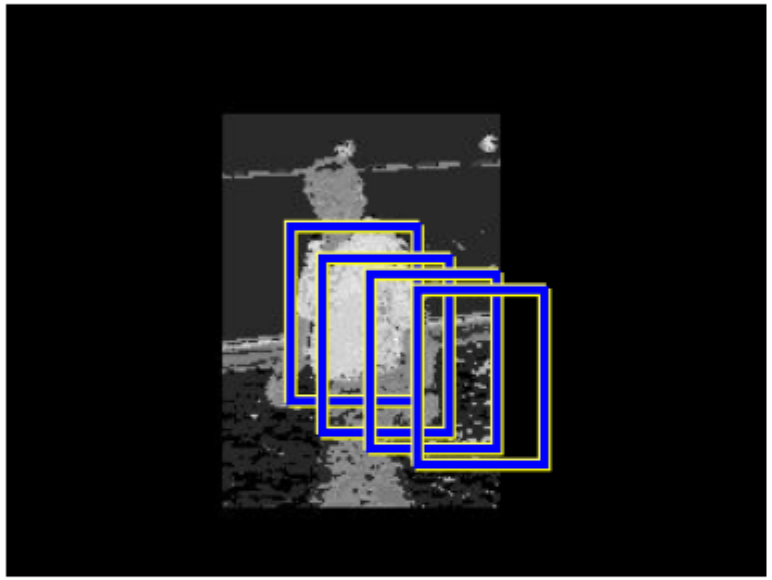
Let pixels form a uniform grid of data points, each with a weight (pixel value) proportional to the "likelihood" that the pixel is on the object we want to track. Perform standard mean-shift algorithm using this weighted set of points.

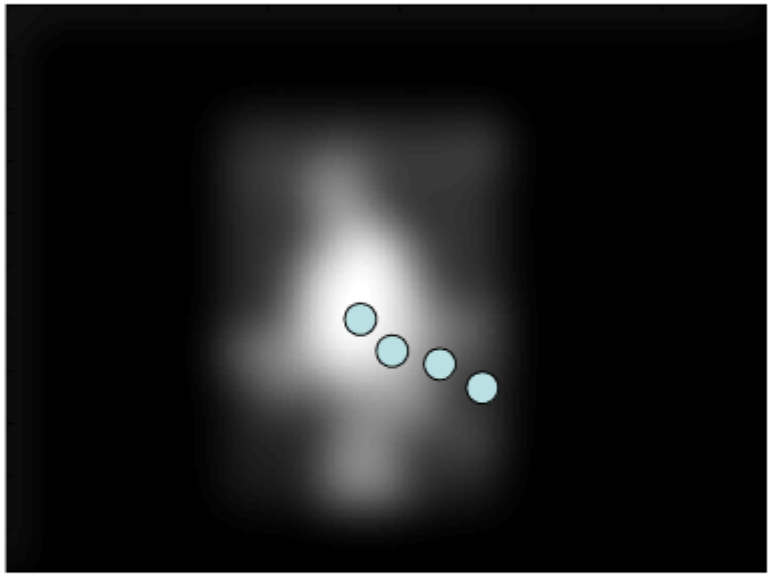


$$\Delta x = \frac{\sum_a K(a-x)\, w(a)\, (a-x)}{\sum_a K(a-x)\, w(a)}$$

Mean-Shift Tracking: Nice property

Running mean-shift with kernel K on weight image w is equivalent to performing gradient ascent in a (virtual) image formed by convolving w with some "shadow" kernel H.





Note: mode we are looking for is mode of location (x,y) likelihood, NOT mode of the color distribution!

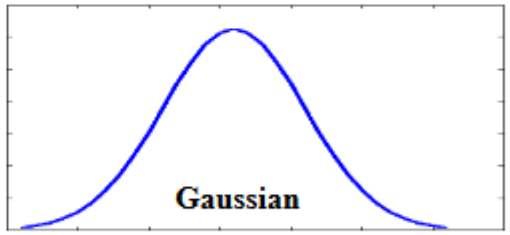
Mean-Shift Tracking: Kernel-Shadow Pairs

Given a convolution kernel H, what is the corresponding mean-shift kernel K?
Perform change of variables $r = \|a-x\|^2$
Rewrite $H(a-x) \Rightarrow h(\|a-x\|^2) \Rightarrow h(r)$.

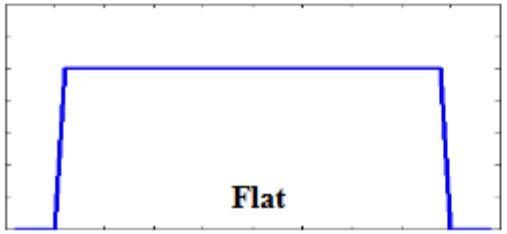
Then kernel K must satisfy **$h'(r) = - c\, k\, (r)$**

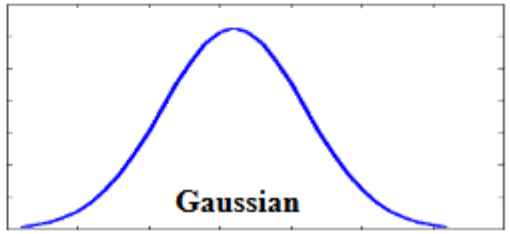
Examples

Shadow



Kernel





Use 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.

Mean-Shift Tracking: Color distribution using histogram model high level overview

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.

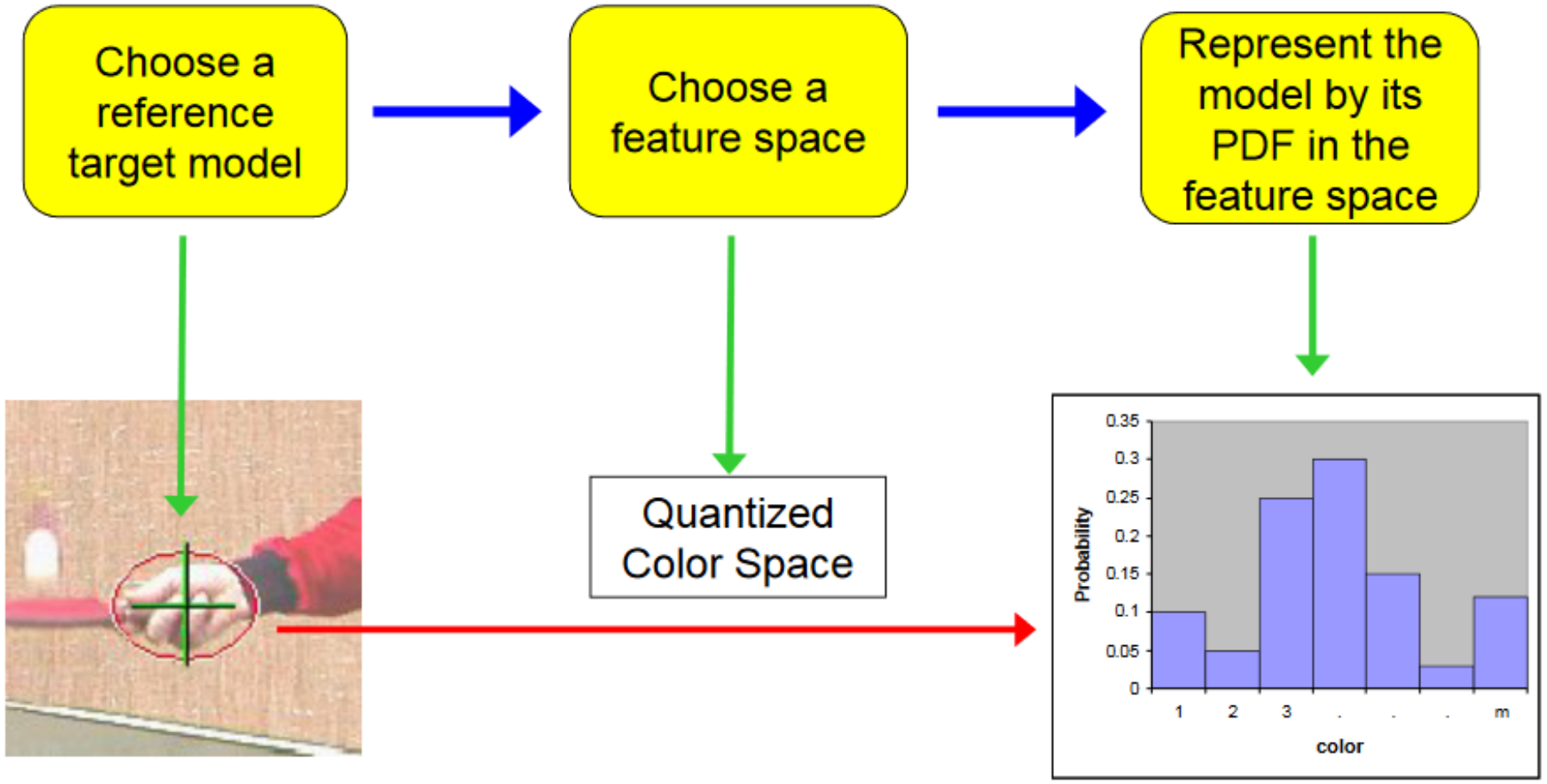
Result is weighted mean shift we used before. However,
the color weights are now computed "on-the-fly", and
change from one iteration to the next.

Mean Shift Object Tracking algorithm

- Object tracking with a Mean-Shift algorithm can be divided into three stages:
  - **Target the object** – In the first frame, we choose the initial location of the object that we want to track. Then we represent the target model with a color histogram. When the object moves, this movement will be reflected in the histogram.
  - **Finding the new location** – In the next frame, the Mean-Shift algorithm moves the window to the new location with maximum pixel density. Now, we use the current histogram to search for the best target match candidate by maximizing the similarity function.
  - **Updating the location** – We update the histogram and the location of the target object.

Mean-Shift Tracking: Target Representation (Color histogram)

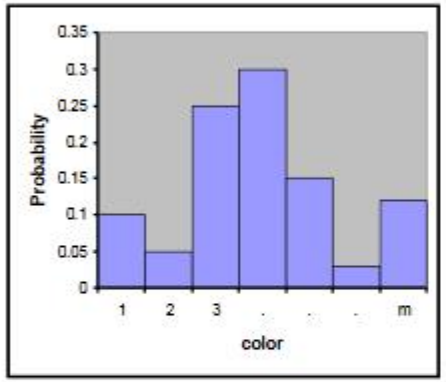
Target Representation (Target Model)

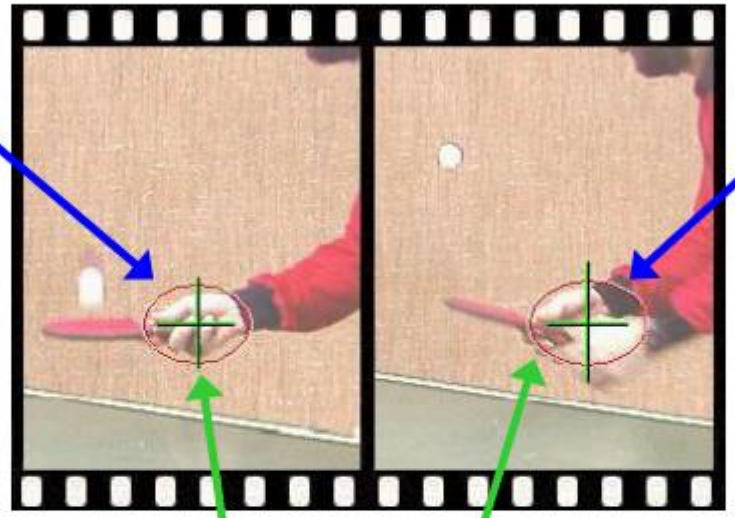


Mean-Shift Tracking: PDF Representation (Color histogram)

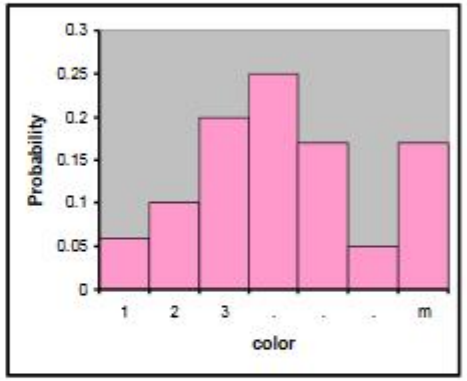
PDF Representation

Target Model
(centered at 0)





Target Candidate
(centered at y)



$$\vec{q} = \{q_u\}_{u=1..m} \qquad \sum_{u=1}^{m} q_u = 1$$

$$\vec{p}(y) = \{p_u(y)\}_{u=1..m} \qquad \sum_{u=1}^{m} p_u = 1$$

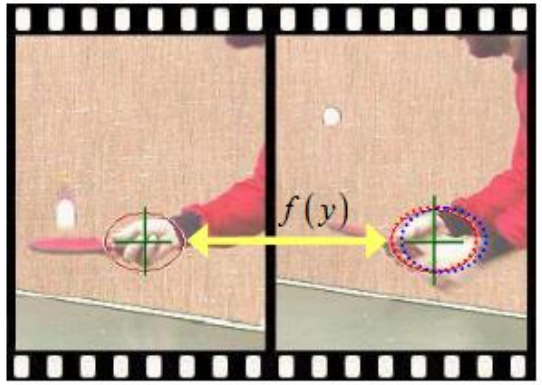
Similarity Function:

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

Mean-Shift Tracking: Smoothness of Similarity Function

Smoothness of Similarity Function

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



Problem:

Target is represented by color info only → Spatial info is lost → Large similarity variations for adjacent locations →

f is not smooth → **Gradient-based optimizations are not robust**

Solution:

Mask the target with an isotropic kernel in the spatial domain → *f*(*y*) becomes smooth in *y*

A Kernel which does not reshape the input image (keeps the same size)

Mean-Shift Tracking: Find the PDF of the target model

Finding the PDF of the target model

$\{x_i\}_{i=1..n}$ Target pixel locations

$k(x)$ A differentiable, isotropic, convex, monotonically decreasing kernel

- Peripheral pixels are affected by occlusion and background interference

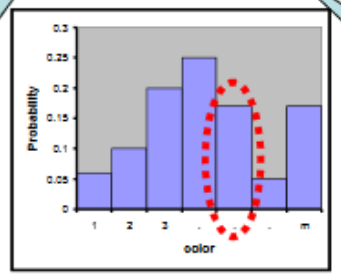
$b(x)$ The color bin index (1..*m*) of pixel *x*

Probability of feature u in model

$$q_u = C \sum_{b(x_i)=u} k\left(\|x_i\|^2\right)$$

Normalization factor

Pixel weight

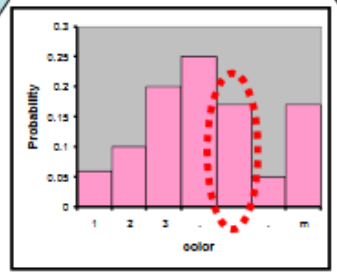


Probability of feature u in candidate

$$p_u(y) = C_h \sum_{b(x_i)=u} k\left(\left\|\frac{y - x_i}{h}\right\|^2\right)$$

Normalization factor

Pixel weight



Mean-Shift Tracking: Similarity Function

Similarity Function

Target model: $\vec{q} = (q_1, \ldots, q_m)$

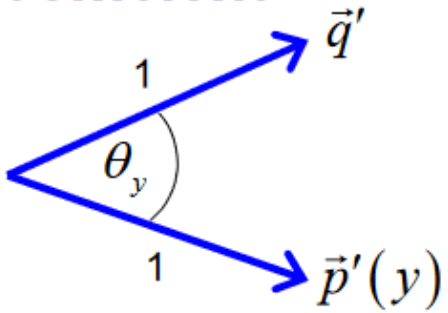
Target candidate: $\vec{p}(y) = (p_1(y), \ldots, p_m(y))$

Similarity function: $f(y) = f[\vec{p}(y), \vec{q}] = ?$

The Bhattacharyya Coefficient

$\vec{q}' = (\sqrt{q_1}, \ldots, \sqrt{q_m})$

$\vec{p}'(y) = (\sqrt{p_1(y)}, \ldots, \sqrt{p_m(y)})$



$$f(y) = \cos\theta_y = \frac{p'(y)^T q'}{\|p'(y)\| \cdot \|q'\|} = \sum_{u=1}^{m} \sqrt{p_u(y) q_u}$$

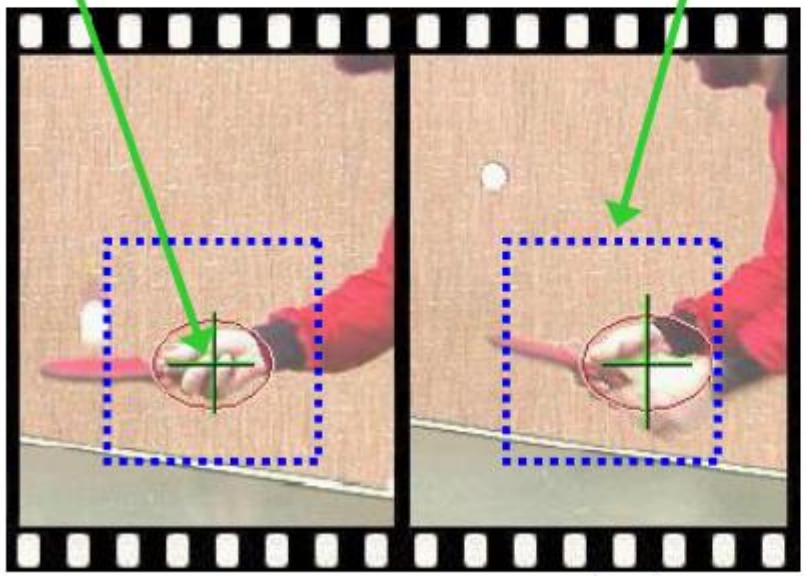
Mean-Shift Tracking: Target Localization Algorithm

Target Localization Algorithm

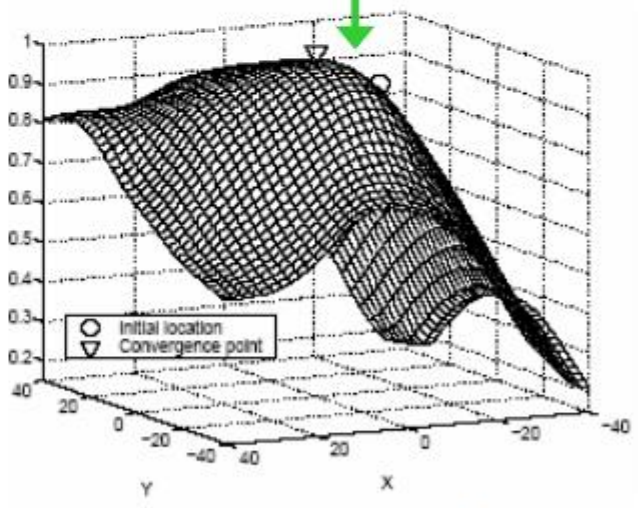
Start from the position of the model in the current frame

Search in the model's neighborhood in next frame

Find best candidate by maximizing a similarity func.

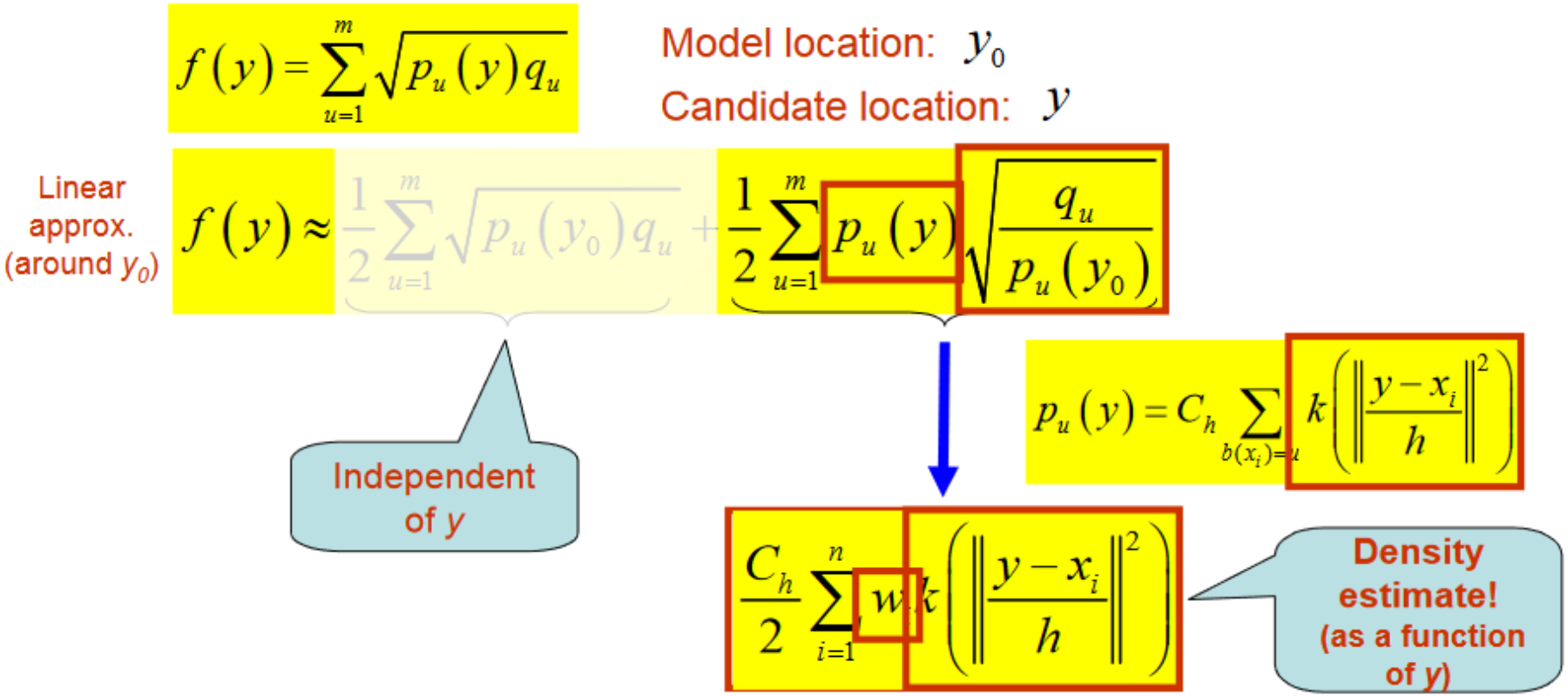


$\vec{q}$ $\vec{p}(y)$



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

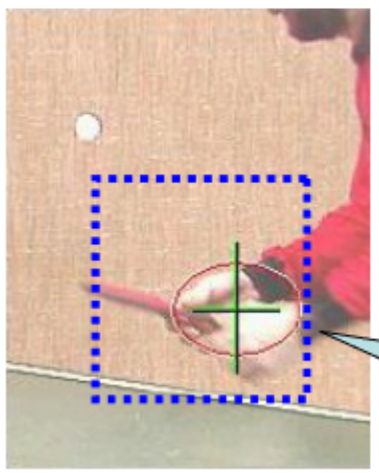
Mean-Shift Object Tracking: Approximating the Similarity Function



Mean-Shift Object Tracking: Maximizing the Similarity Function

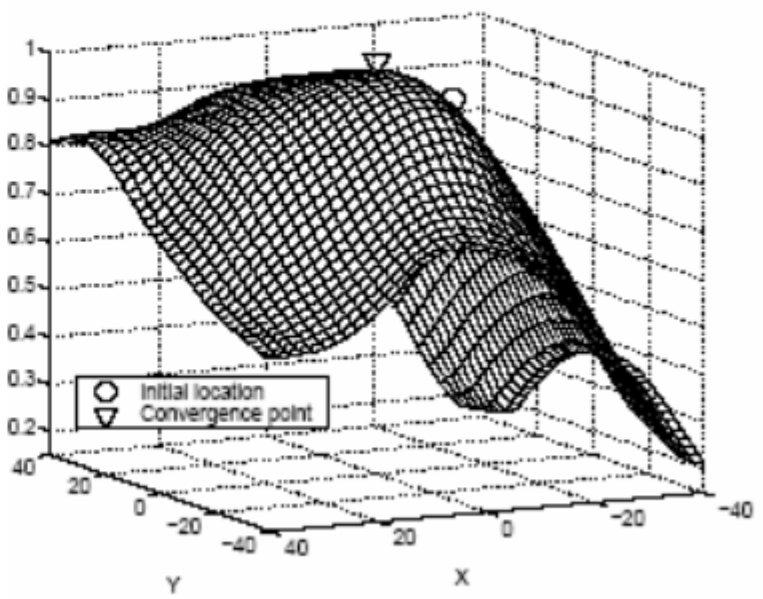
The mode of $\frac{C_h}{2}\sum_{i=1}^{n} w_i k\left(\left\|\frac{y-x_i}{h}\right\|^2\right)$ = sought maximum

Important Assumption:



The target representation provides sufficient discrimination

One mode in the searched neighborhood



Mean-Shift Object Tracking: Applying Mean Shift

Applying Mean-Shift

The mode of $\frac{C_h}{2}\sum_{i=1}^{n} w_i k\left(\left\|\frac{y-x_i}{h}\right\|^2\right)$ = sought maximum

Original Mean-Shift:

Find mode of $c\sum_{i=1}^{n} k\left(\left\|\frac{y-x_i}{h}\right\|^2\right)$ using $y_1 = \frac{\sum_{i=1}^{n} x_i g\left(\left\|\frac{y_0-x_i}{h}\right\|^2\right)}{\sum_{i=1}^{n} g\left(\left\|\frac{y_0-x_i}{h}\right\|^2\right)}$

Extended Mean-Shift:

Find mode of $c\sum_{i=1}^{n} w_i k\left(\left\|\frac{y-x_i}{h}\right\|^2\right)$ using $y_1 = \frac{\sum_{i=1}^{n} x w_i g\left(\left\|\frac{y_0-x_i}{h}\right\|^2\right)}{\sum_{i=1}^{n} w_i g\left(\left\|\frac{y_0-x_i}{h}\right\|^2\right)}$

Mean-Shift Object Tracking: Applying Mean Shift (About Profiles)

About Kernels and Profiles

A special class of radially symmetric kernels:

$$K(x) = ck\left(\|x\|^2\right)$$

The profile of kernel *K*

$$k'(x) = -g(x)$$

Extended Mean-Shift:

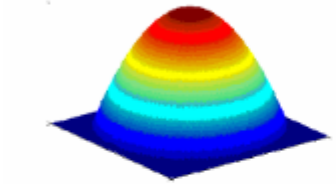
Find mode of $c\sum_{i=1}^{n} w_i k\left(\left\|\frac{y - x_i}{h}\right\|^2\right)$ using $y_1 = \frac{\sum_{i=1}^{n} x_i w_i g\left(\left\|\frac{y_0 - x_i}{h}\right\|^2\right)}{\sum_{i=1}^{n} w_i g\left(\left\|\frac{y_0 - x_i}{h}\right\|^2\right)}$

Mean-Shift Object Tracking: Applying Mean Shift (choosing the kernel)

A special class of radially symmetric kernels:

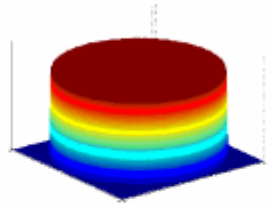
$$K(x) = ck\left(\|x\|^2\right)$$

Epanechnikov kernel:



$$k(x) = \begin{cases} 1-x & \text{if } \|x\| \le 1 \\ 0 & \text{otherwise} \end{cases}$$

Uniform kernel:



$$g(x) = -k(x) = \begin{cases} 1 & \text{if } \|x\| \le 1 \\ 0 & \text{otherwise} \end{cases}$$

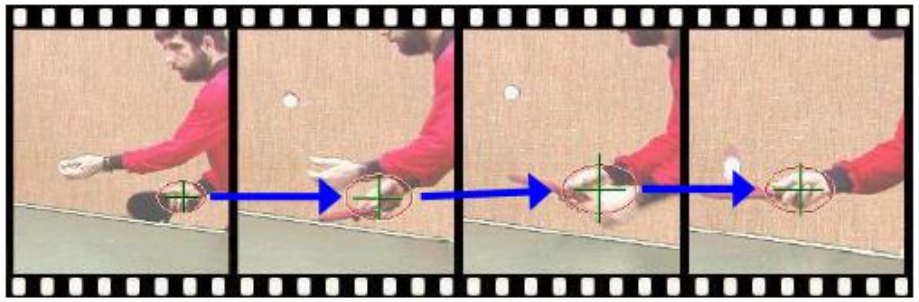
$$y_1 = \frac{\sum_{i=1}^{n} x_i w_i g\left(\left\|\frac{y_0 - x_i}{h}\right\|^2\right)}{\sum_{i=1}^{n} w_i g\left(\left\|\frac{y_0 - x_i}{h}\right\|^2\right)} \rightarrow y_1 = \frac{\sum_{i=1}^{n} x_i w_i}{\sum_{i=1}^{n} w_i}$$

Mean-Shift Object Tracking – Adaptive Scale

Adaptive Scale

Problem:

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



Solution:

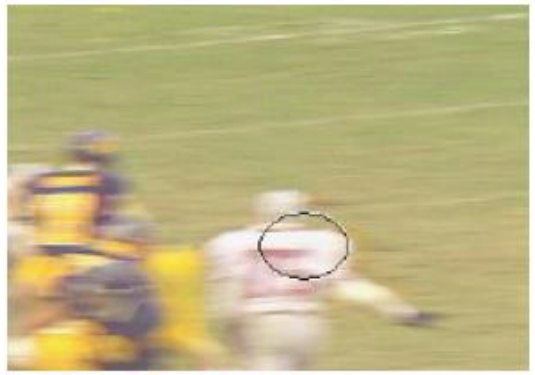
Run localization 3 times with different *h* → Choose *h* that achieves maximum similarity

Results

From Comaniciu, Ramesh, Meer

Partial occlusion

Distraction



Motion blur

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

References

- Mean Shift in Space by Udacity
- https://www.comp.nus.edu.sg/~cs4243/lecture/meanshift.pdf
- https://www.cse.psu.edu/~rtc12/CSE598G/introMeanShift.pdf
- https://campar.in.tum.de/twiki/pub/Chair/TeachingWs12TDCV/mean_shift.pdf
- https://www.cs.cmu.edu/~16385/s17/Slides/15.2_Tracking__Mean_Shift.pdf
- https://datahacker.rs/object-tracking-with-mean-shift-and-camshift-algorithms/
- Mean-Shift Segmentation | Image Segmentation by Shree K Nayar_Columbia University
- https://openaccess.thecvf.com/content/CVPR2021/papers/Jang_MeanShift_Extremely_Fast_Mode-Seeking_With_Applications_to_Segmentation_and_Object_CVPR_2021_paper.pdf
- CV3DST - computer Vision 3: Detection, Segmentation, and Tracking - Technical University of Munich - Prof. Leal-Taixé (WS21/22)