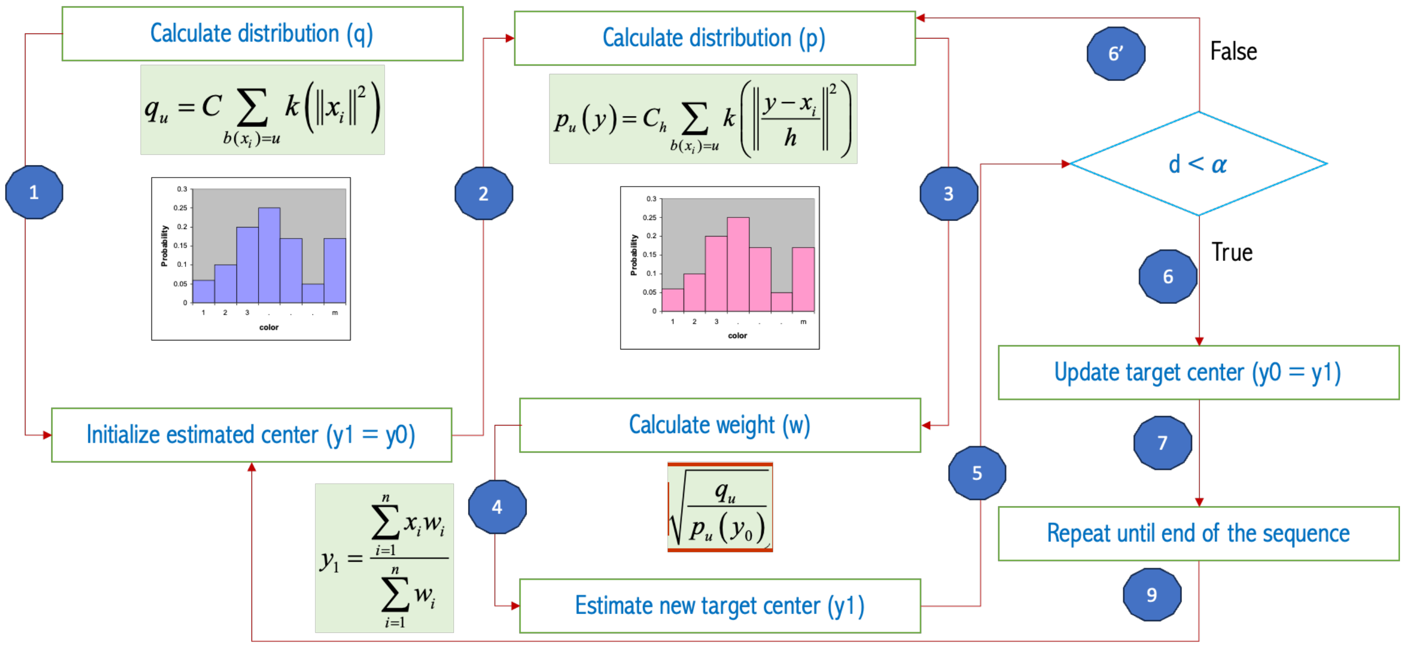
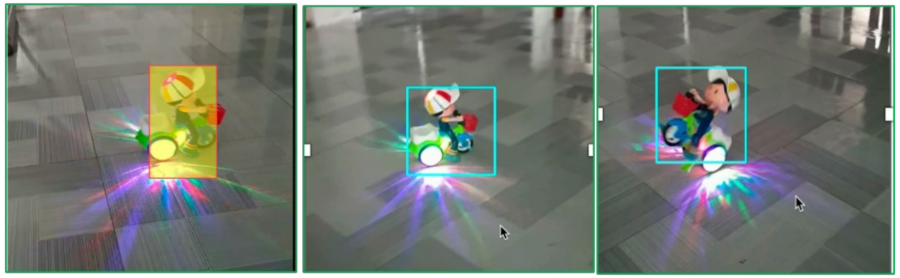
AI VIETNAM
All-in-One Course

Object Tracking: Mean-Shift

(Computer Vision Foundation)





Frame t-2 Frame t-1 Frame t

Vinh Dinh Nguyen
PhD in Computer Science

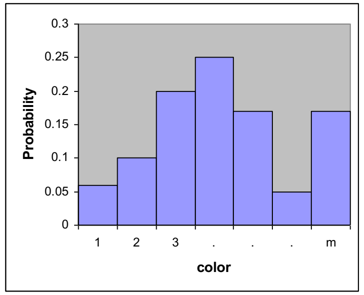
Year 2023

Mean Shift Overview

1

Calculate distribution (q)

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

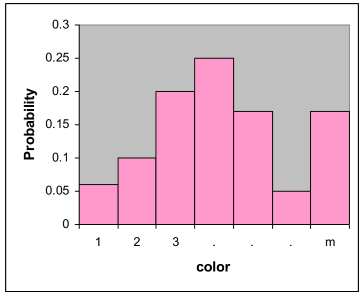


Initialize estimated center (y1 = y0)

2

Calculate distribution (p)

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



3

Calculate weight (w)

$$\sqrt{\frac{q_u}{p_u(y_0)}}$$

4

Estimate new target center (y1)

$$y_1 = \frac{\sum_{i=1}^{n} x_i w_i}{\sum_{i=1}^{n} w_i}$$

5

$d < \alpha$

6' False

6 True

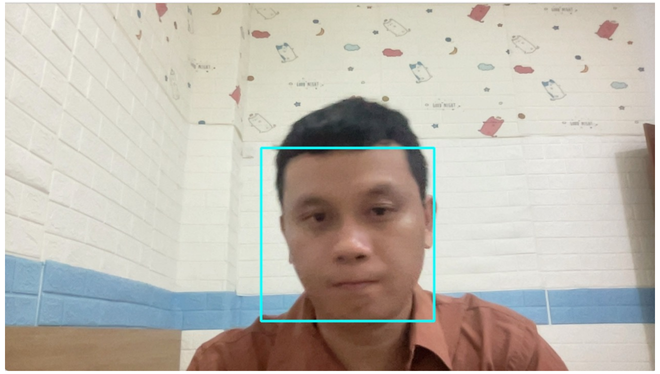
Update target center (y0 = y1)

7

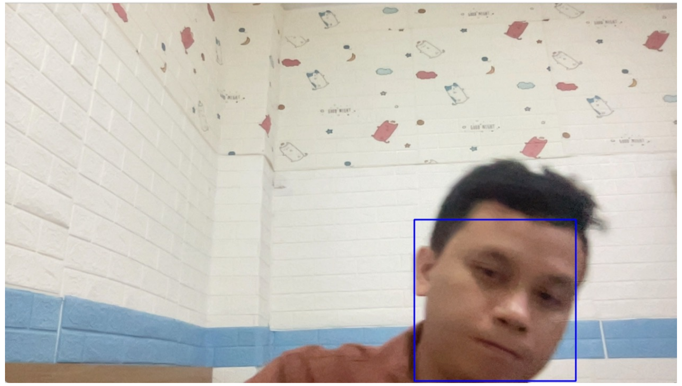
Repeat until end of the sequence

9

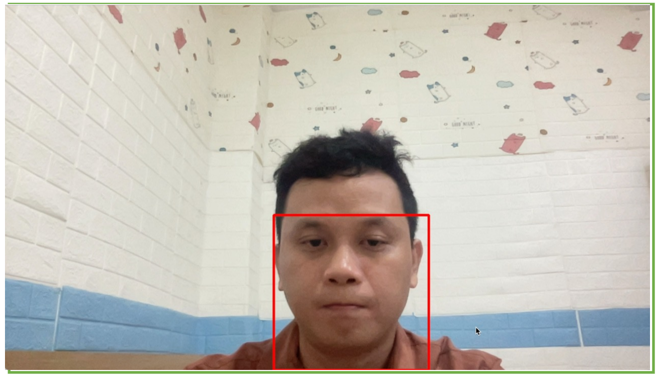
Face Detection and Tracking: Static Background



Face detection without tracking
(video)

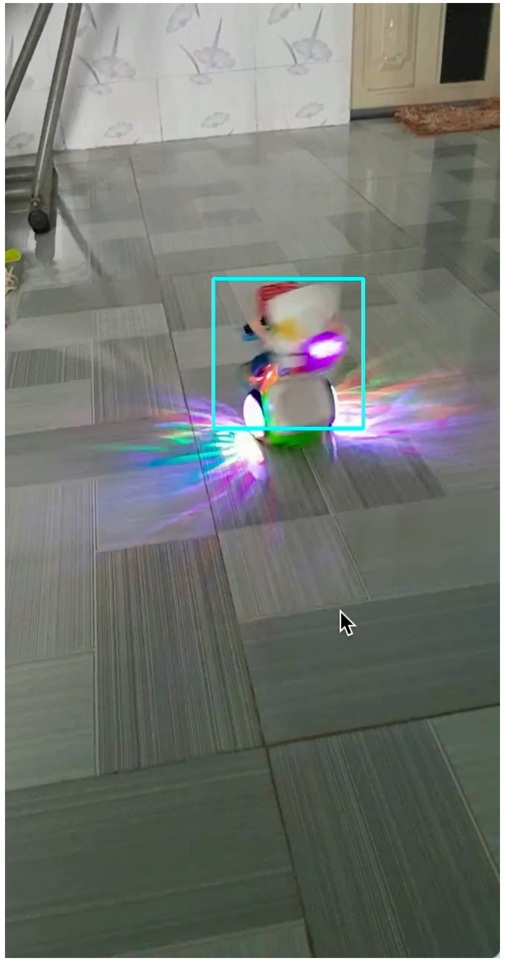


Face detection with tracking-based template
(video)

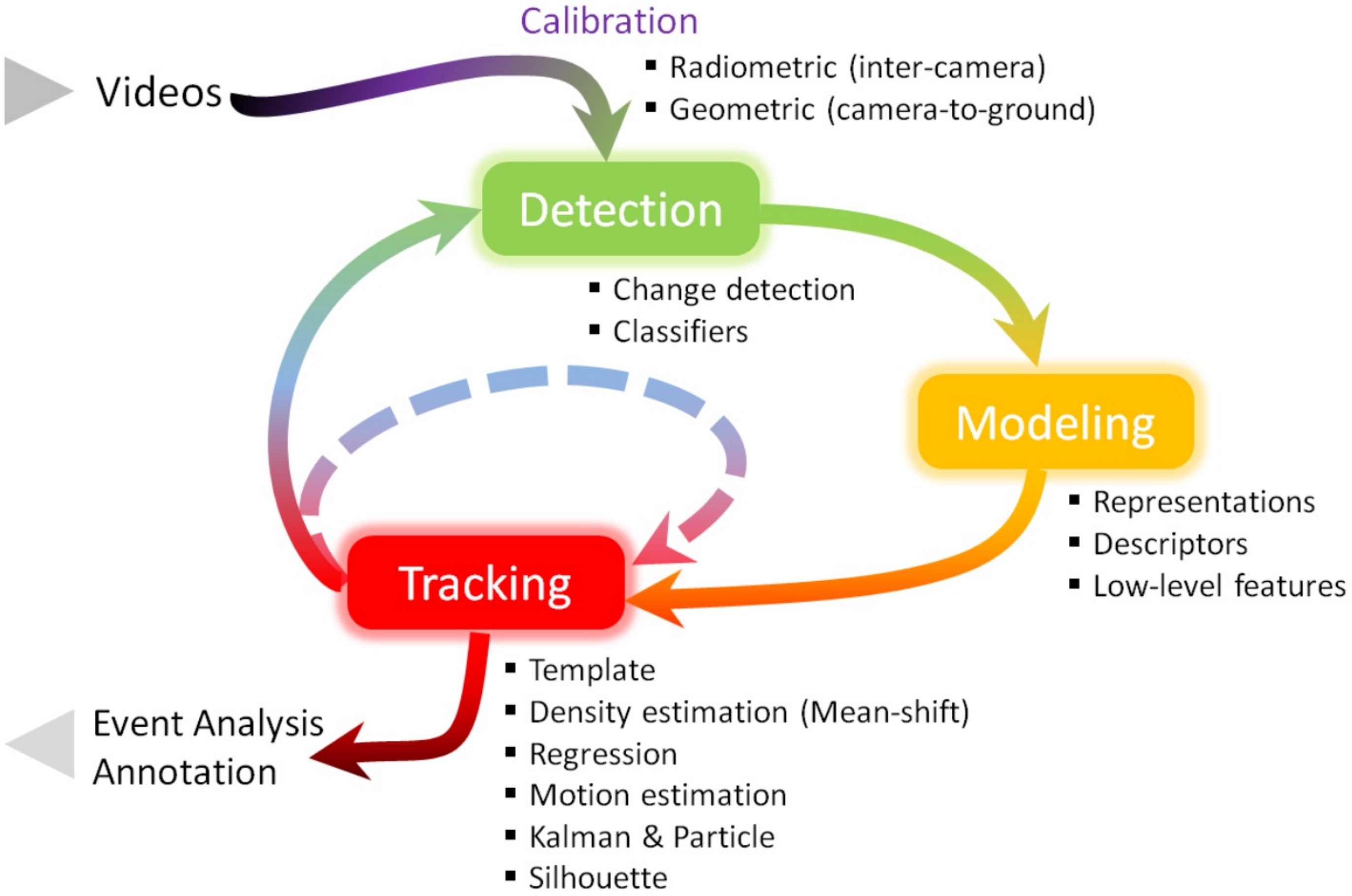


Face detection with tracking-based meanshift
(video)

Face Detection and Tracking: Dynamic Background



Object detection & tracking (video)



Outline

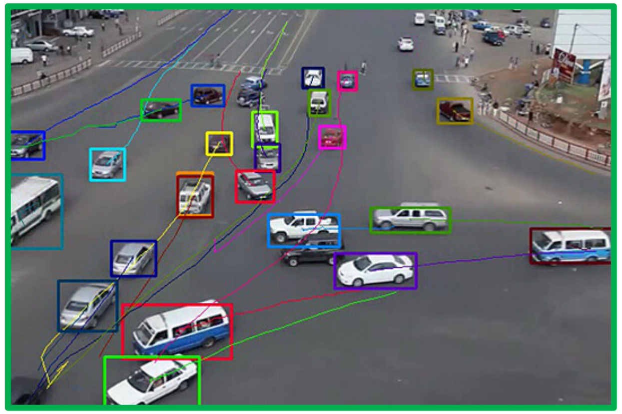
- Object Tracking and Its Application
- Object Tracking Based on Template Matching
- Object Tracking Based on Histogram Matching
- Mean Shift Motivation
- Mean Shift Object Tracking
- Object Tracking by SIFT feature
- Face Detection and Tracking

Outline

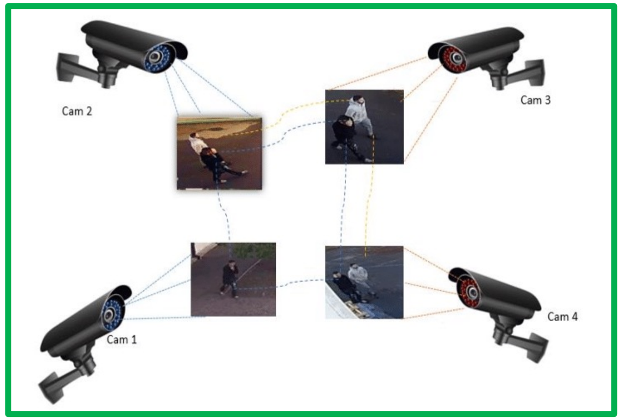
- Object Tracking and Its Application
- Object Tracking Based on Template Matching
- Object Tracking Based on Histogram Matching
- Mean Shift Motivation
- Mean Shift Object Tracking
- Object Tracking by SIFT feature
- Face Detection and Tracking

Tracking Applications

Suppermarket



Traffic monitoring



Multicamera object tracking

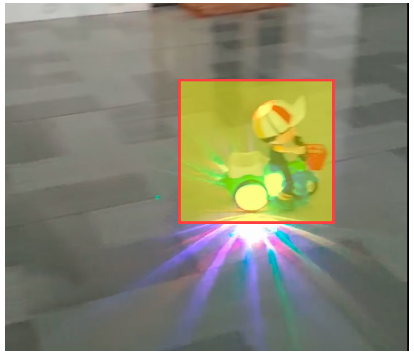
Other applications

Outline

- Object Tracking and Its Application
- Object Tracking Based on Template Matching
- Object Tracking Based on Histogram Matching
- Mean Shift Motivation
- Mean Shift Object Tracking
- Object Tracking by SIFT feature
- Face Detection and Tracking

Object Tracking Based on Template

Given: location of target in initial or previous frame
Find: location of target in current frame



Input image

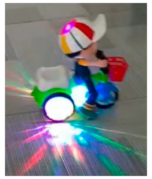
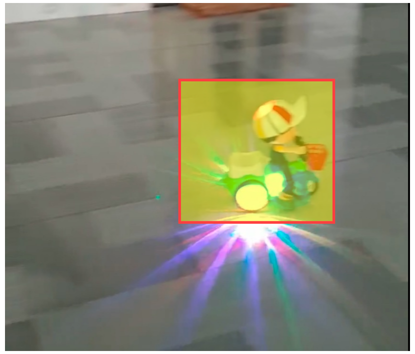
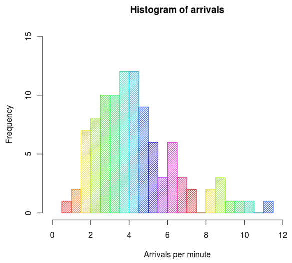


Image Template

Appeareance based Tracking



Input image



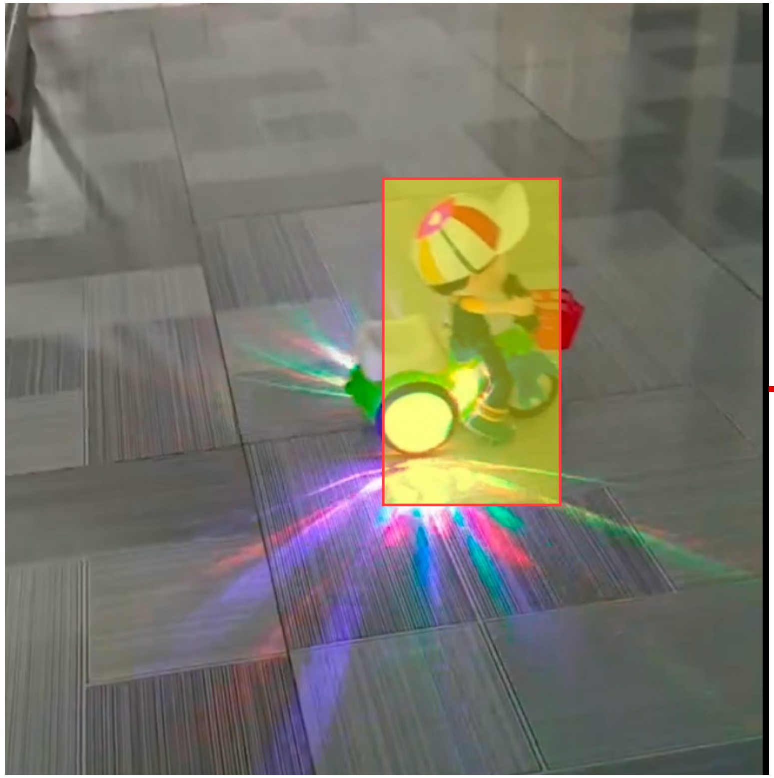
Histogram Template

Histogram based Tracking

Tracking using Appearance Matching

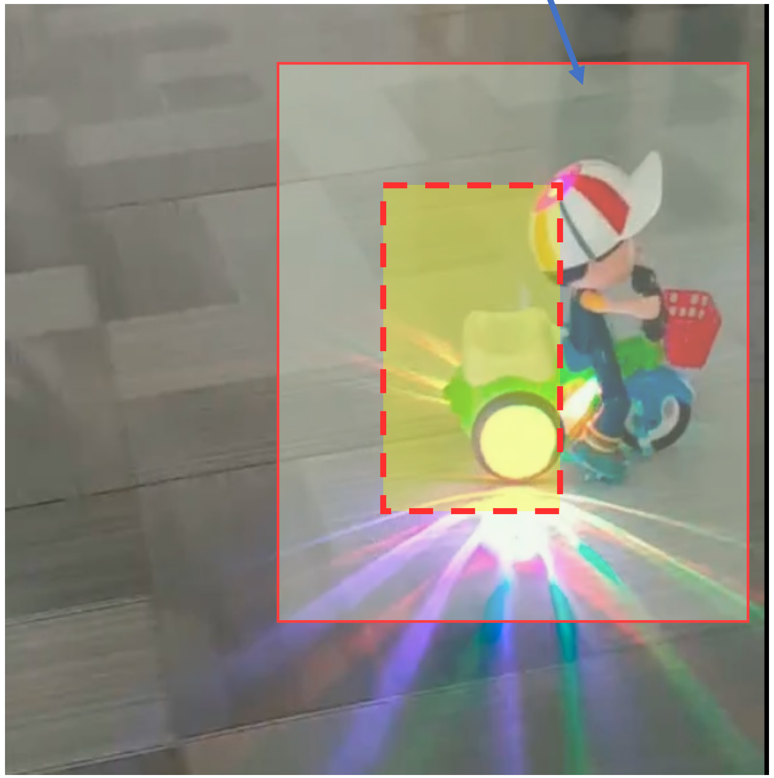
Given template window S in frame (t-1), search neigborhood regions to find match in frame t

Neighborhood regions



Frame t-1

Object Template



Frame t

Simple implementation
Fail in changing scales, viewpoint, occlusion

Similarity Metrics for Template Matching

Sum of absolute Differences

$$SAD(k,l) = \sum_{(i,j)\in T} |I_1(i,j) - I_2(i+k, j+l)|$$

Sum of square Differences

$$SSD(k,l) = \sum_{(i,j)\in T} |I_1(i,j) - I_2(i+k, j+l)|^2$$

Correlation

$$r = \frac{n(\Sigma xy) - (\Sigma x)(\Sigma y)}{\sqrt{[n\Sigma x^2 - (\Sigma x)^2][n\Sigma y^2 - (\Sigma y)^2]}}$$

Similarity Metrics for Template Matching

Cosine Similarity

$$\cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}}$$

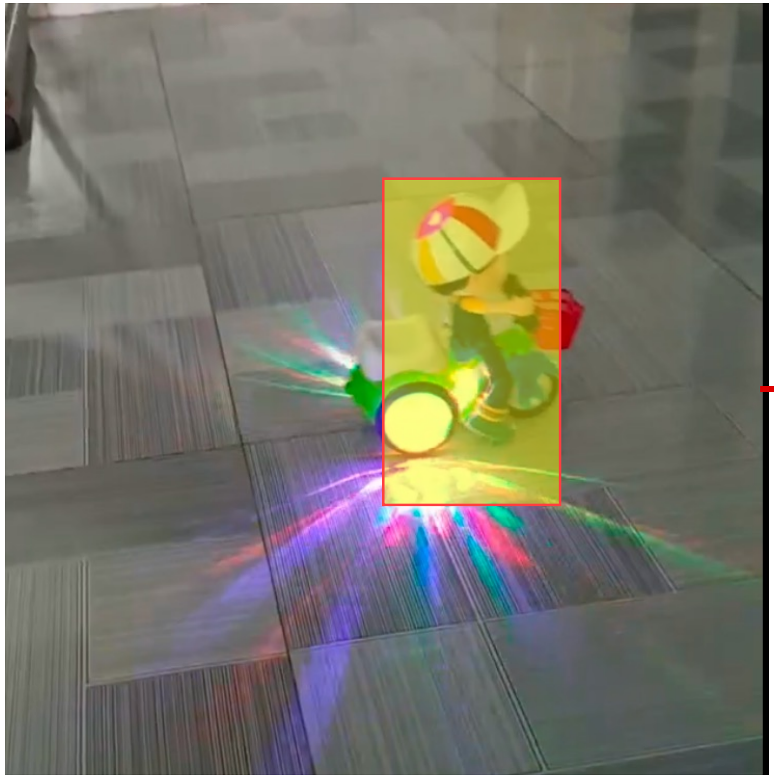
Normalized cross-correlation

$$NCC(k,l) = \frac{\sum_{(i,j)\in T} I_1(i,j)\, I_2(i+k, j+l)}{\sqrt{\sum_{(i,j)\in T} I_1(i,j)^2 \sum_{(i,j)\in T} I_2(i+k, j+l)^2}}$$

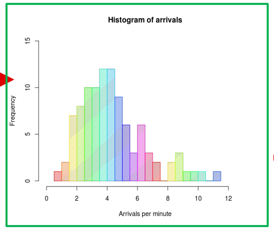
Outline

- Object Tracking and Its Application
- Object Tracking Based on Template Matching
- Object Tracking Based on Histogram Matching
- Mean Shift Motivation
- Mean Shift Object Tracking
- Object Tracking by SIFT feature
- Face Detection and Tracking

Tracking using Histogram Matching

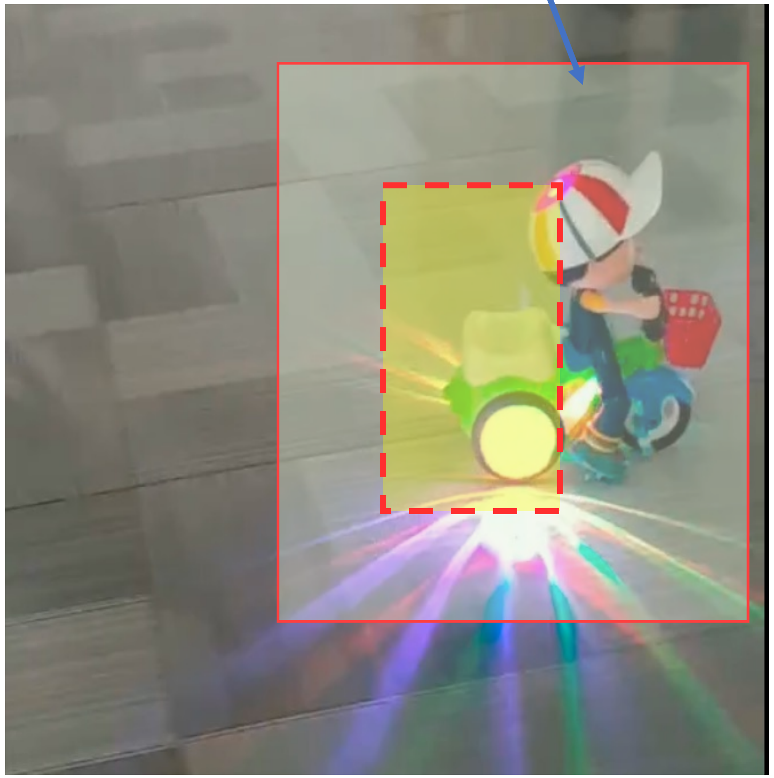


Frame t-1



Histogram Template

Neighborhood regions



Frame t

Comparing Histogram:
Correlation, Square Difference, Histogram Intersection,…

Limitations

Outline

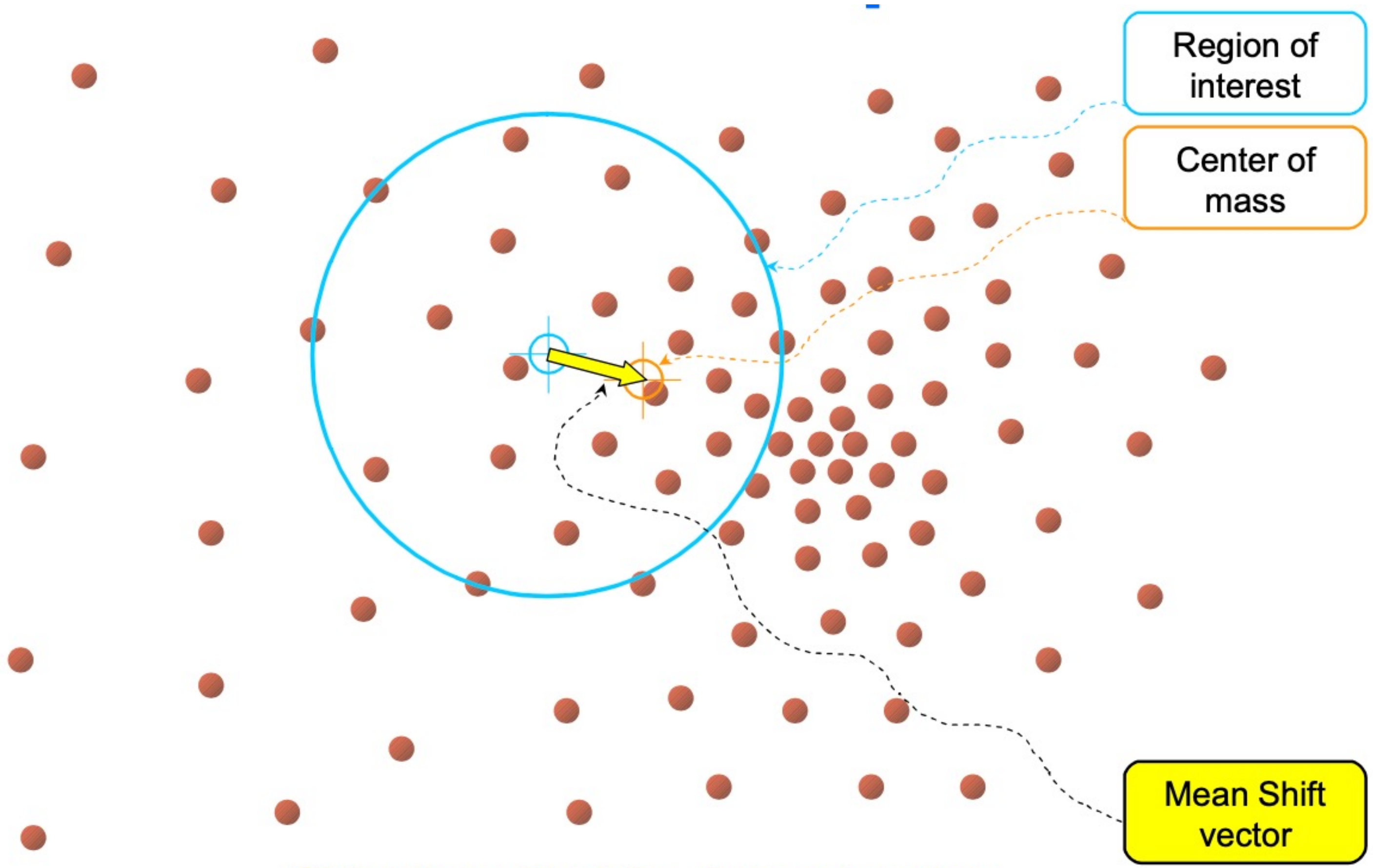
- Object Tracking and Its Application
- Object Tracking Based on Template Matching
- Object Tracking Based on Histogram Matching
- Mean Shift Motivation
- Mean Shift Object Tracking
- Object Tracking by SIFT feature
- Face Detection and Tracking

Mean Shift Motivation

The mean-shift algorithm is an efficient approach to tracking objects whose appearance is defined by histograms

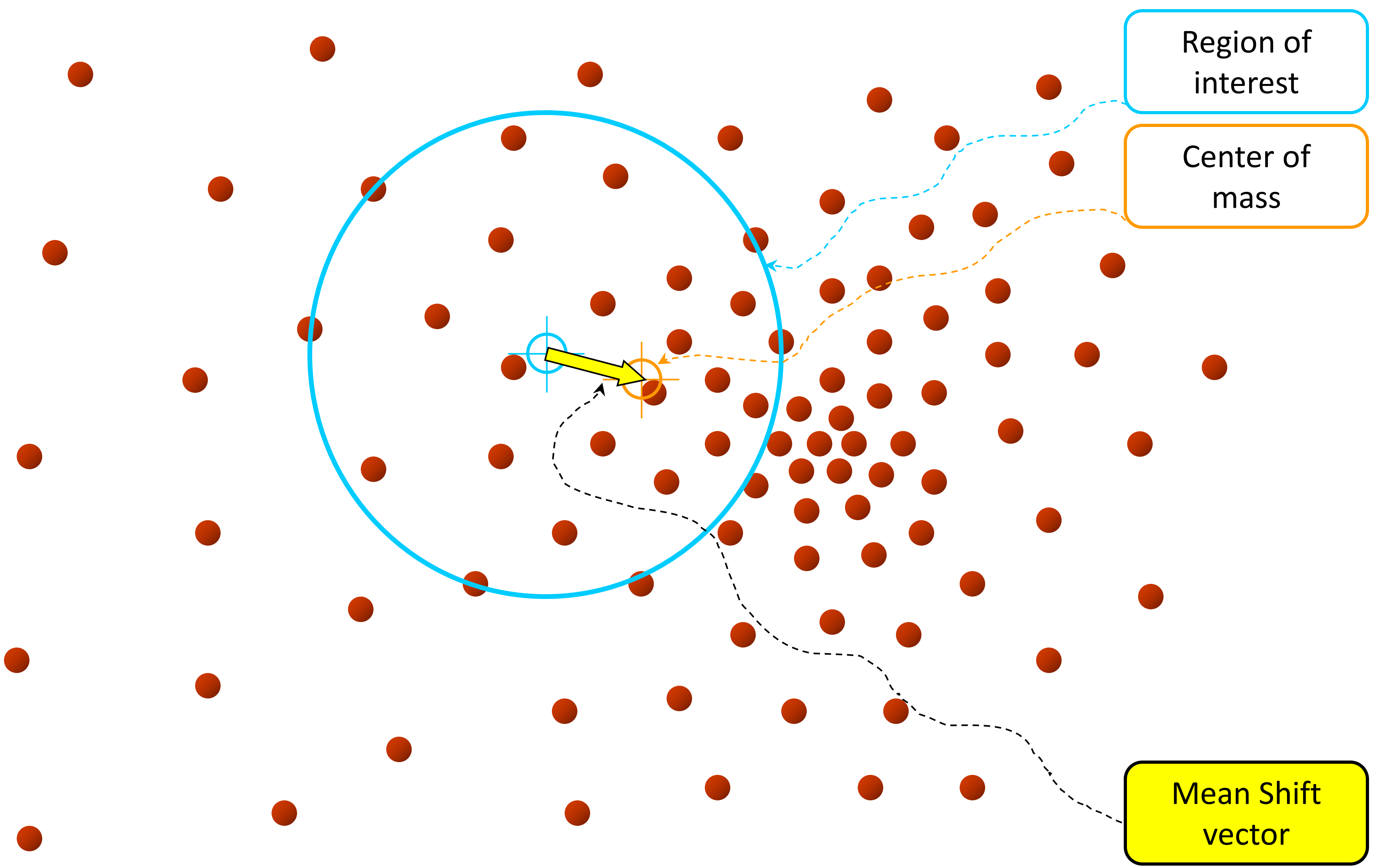
Motivation – to track non-rigid objects, (like a walking person), it is hard to specify an explicit 2D parametric motion model.

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



Objective : Find the densest region

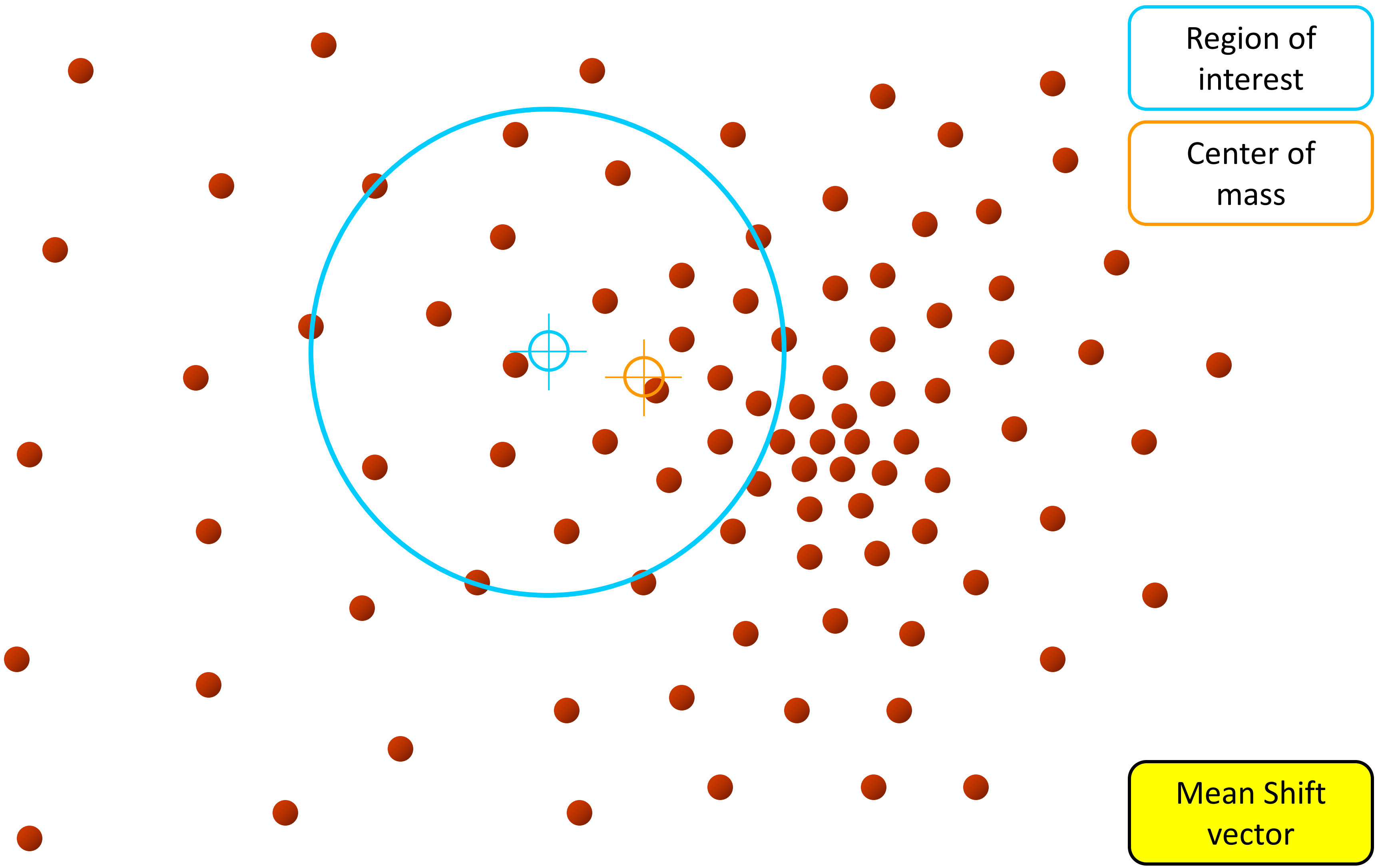
Mean Shift Motivation



Objective : Find the densest region

Distribution of identical billiard balls

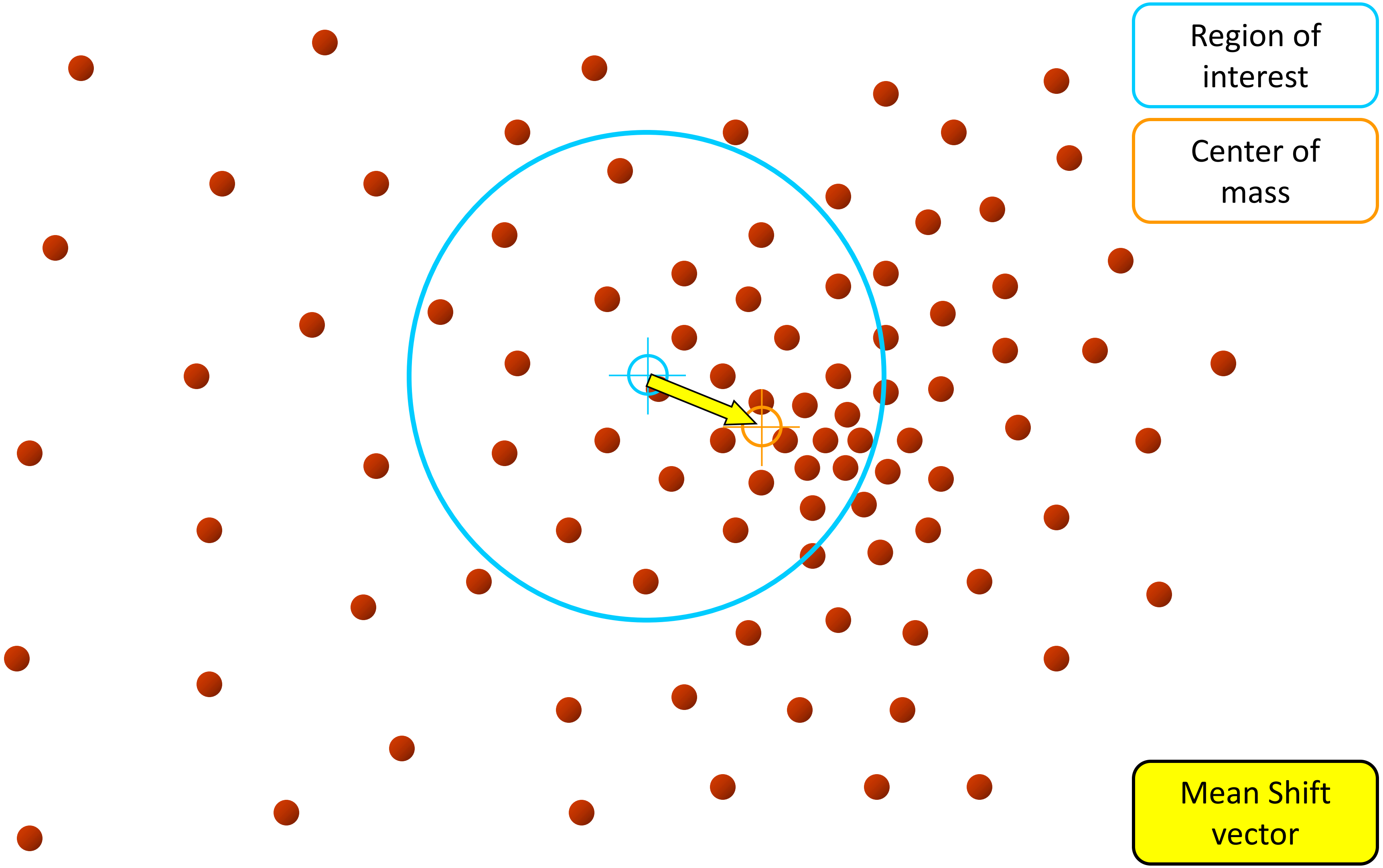
Mean Shift Motivation



Objective : Find the densest region

Distribution of identical billiard balls

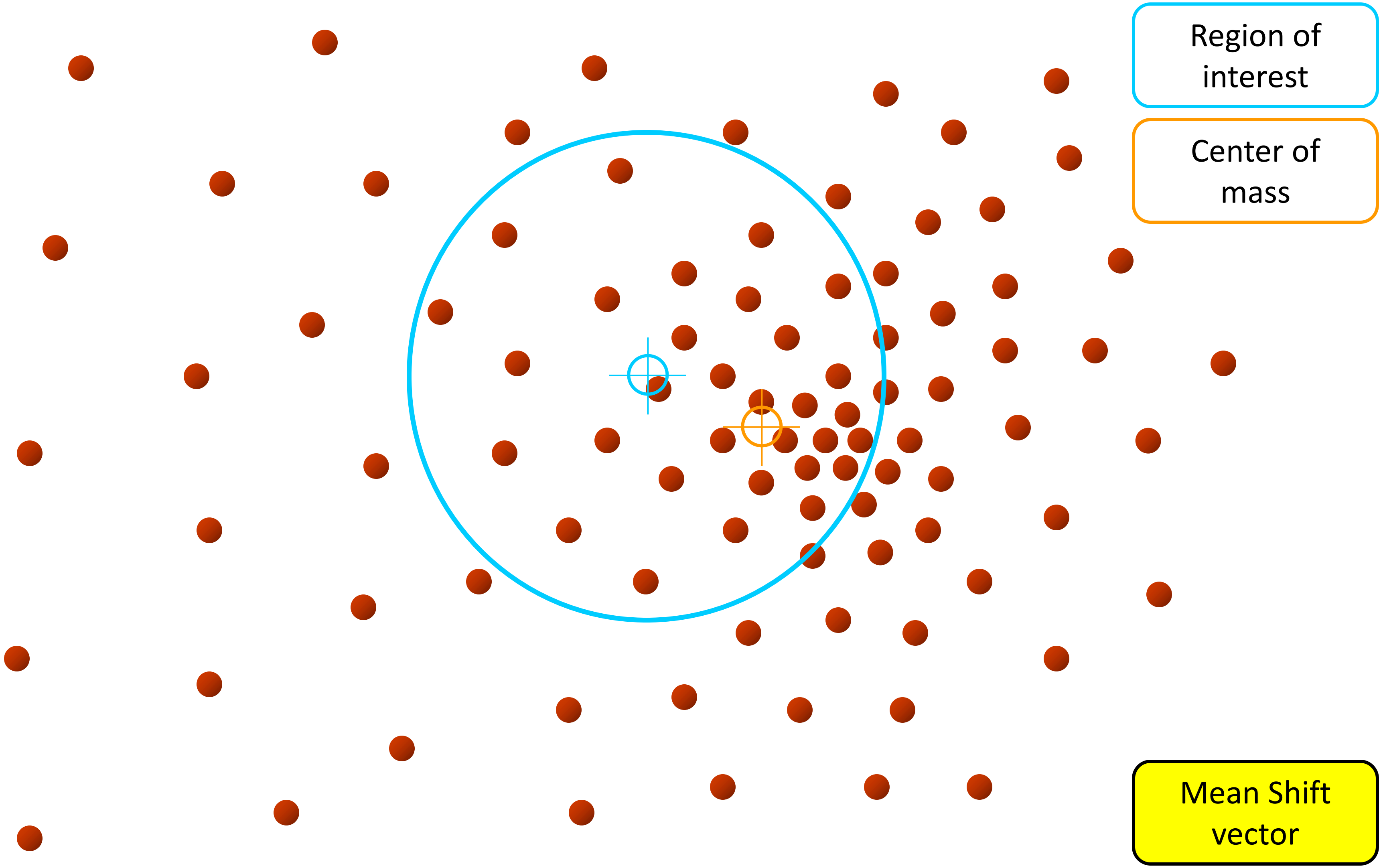
Mean Shift Motivation



Objective : Find the densest region

Distribution of identical billiard balls

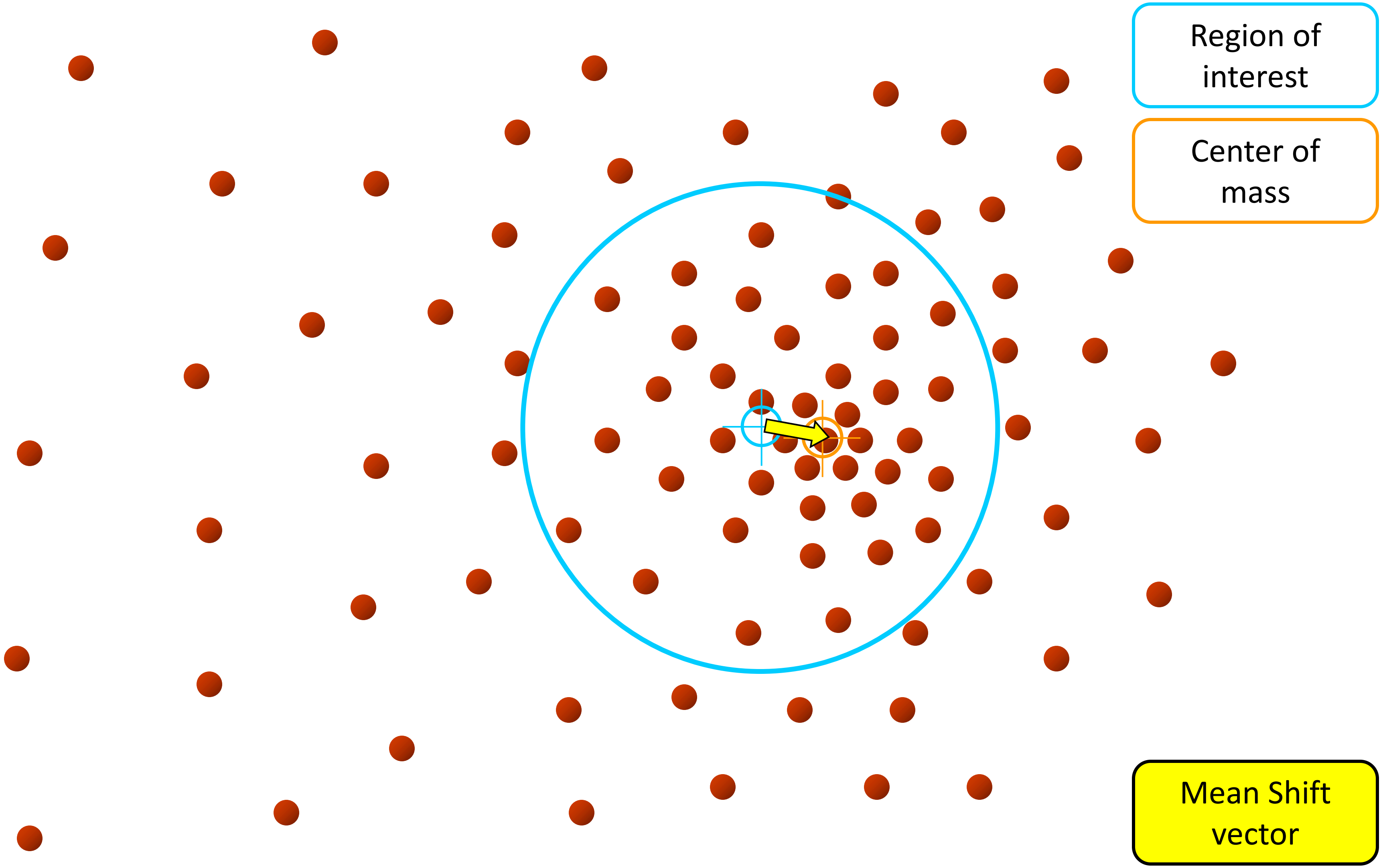
Mean Shift Motivation



Objective : Find the densest region

Distribution of identical billiard balls

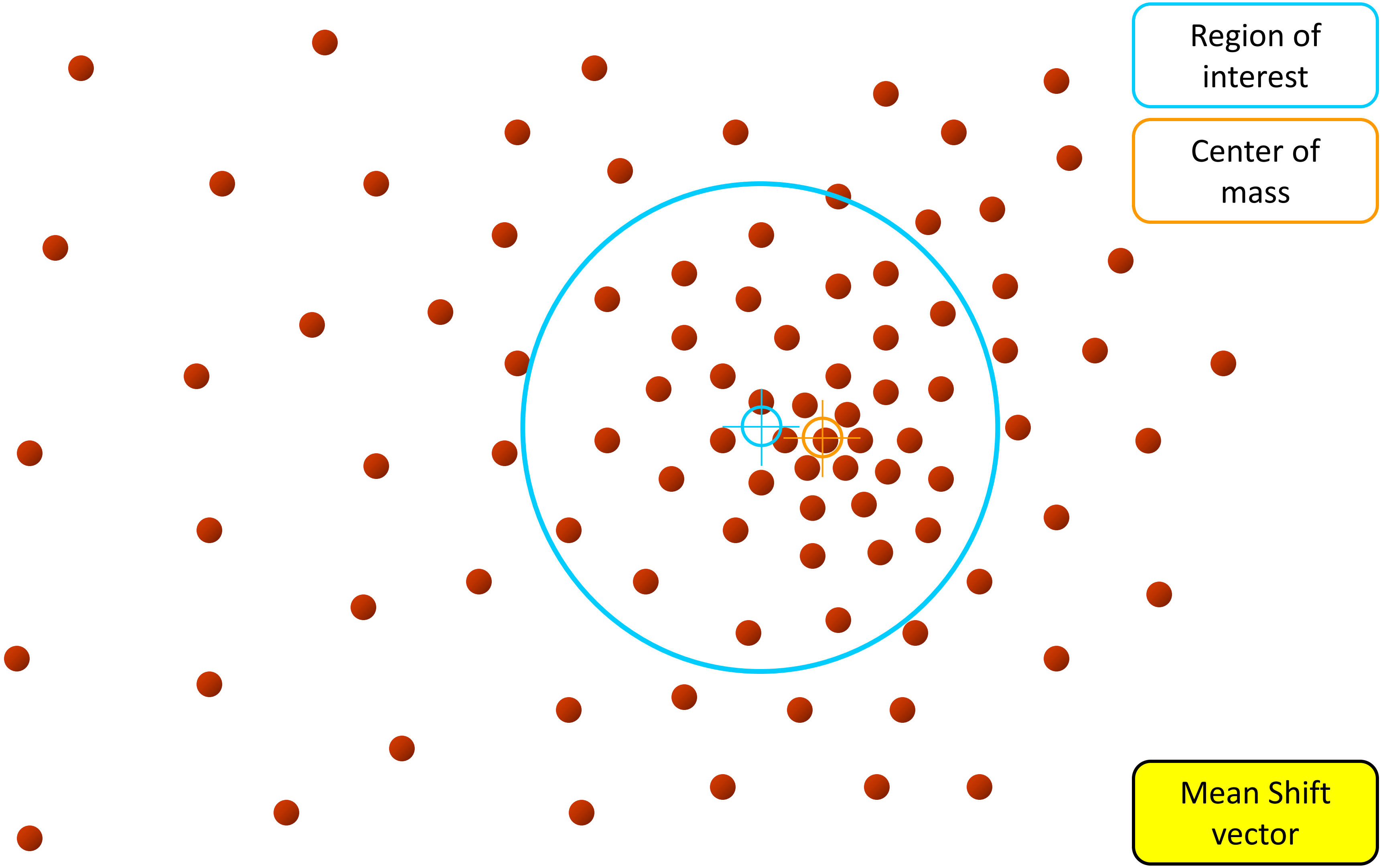
Mean Shift Motivation



Objective : Find the densest region

Distribution of identical billiard balls

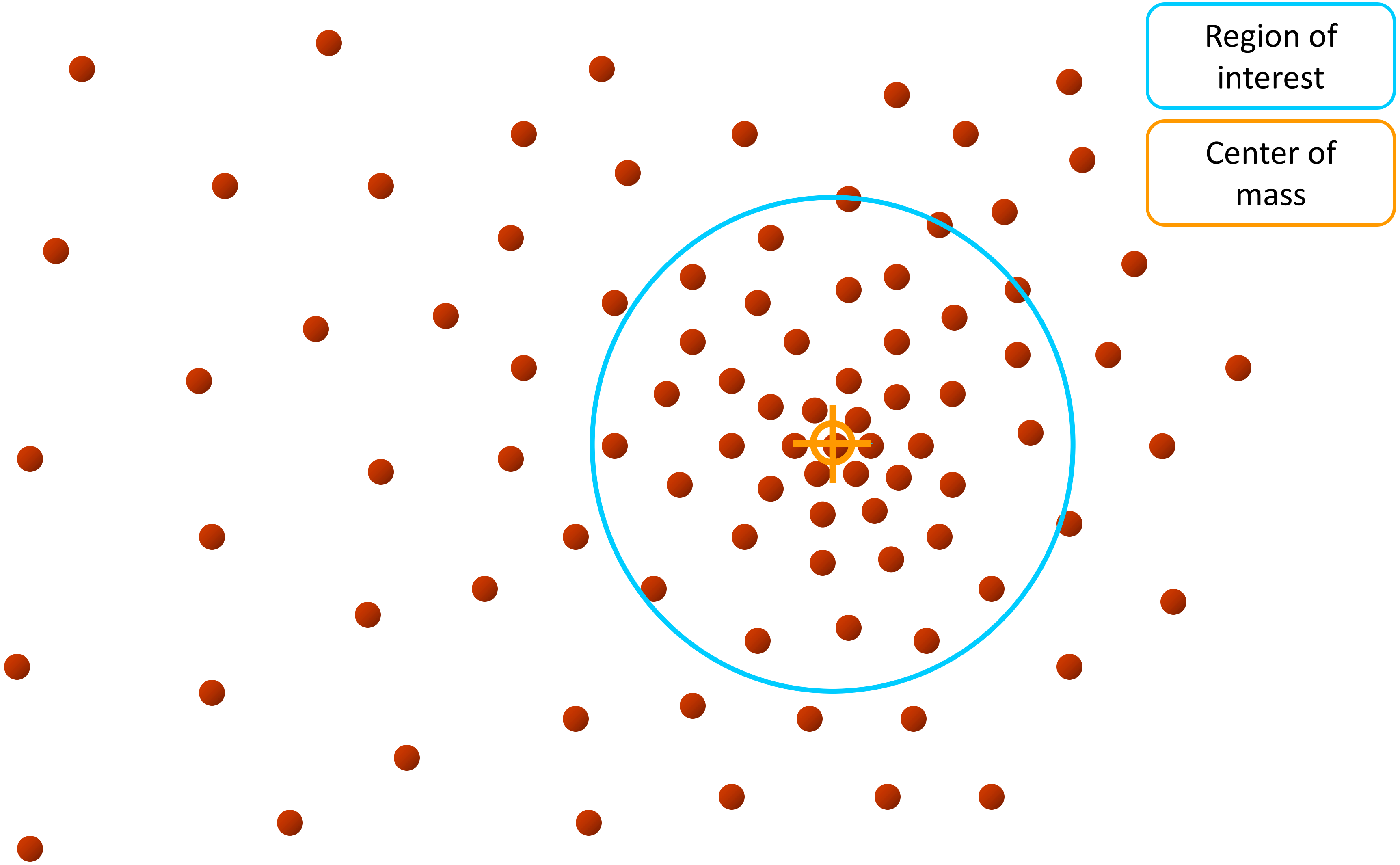
Mean Shift Motivation



Objective : Find the densest region

Distribution of identical billiard balls

Mean Shift Motivation

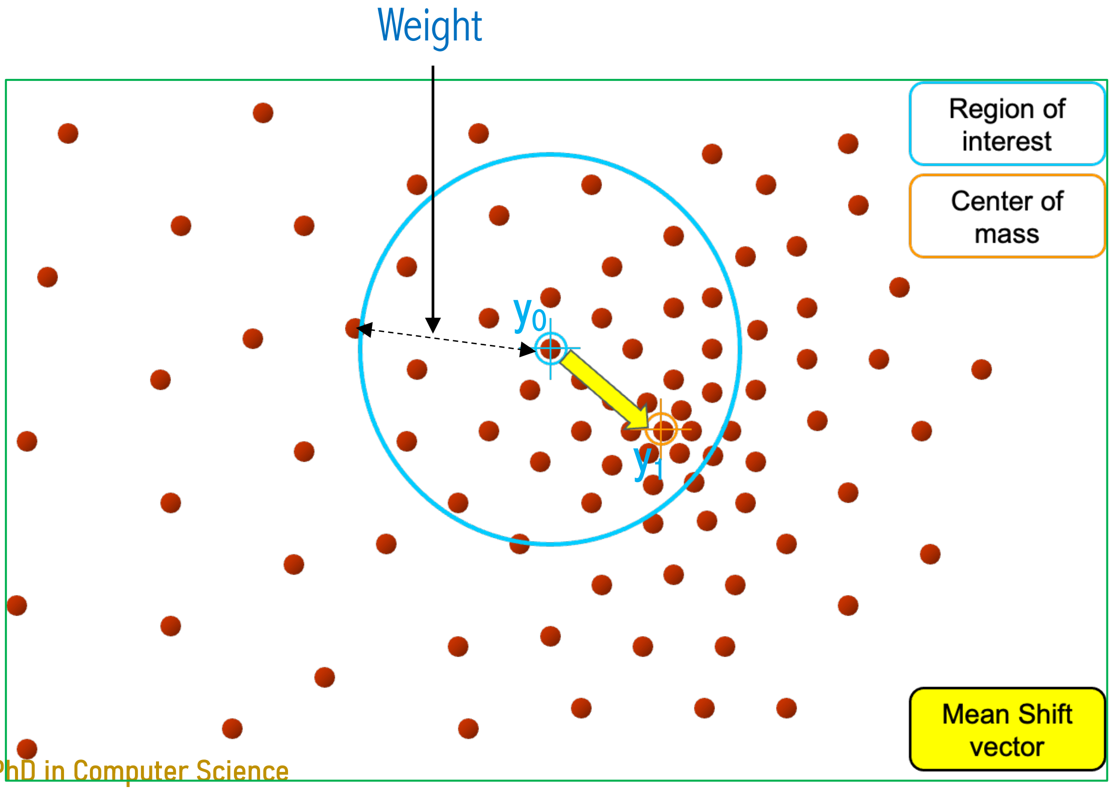


Objective : Find the densest region

Distribution of identical billiard balls

Mean Shift Vector

- Given: data points and approximate location of the mean of this data
- Task: Estimate the exact location of the mean of the data by determining the shift vector from the initial mean.
- Properties of mean shift :
  - Always point towards the direction of the maximum increase in the density
  - Mean shift vector has the direction of the gradient of the density estimate



Mean shift vector: How the mean is shifting

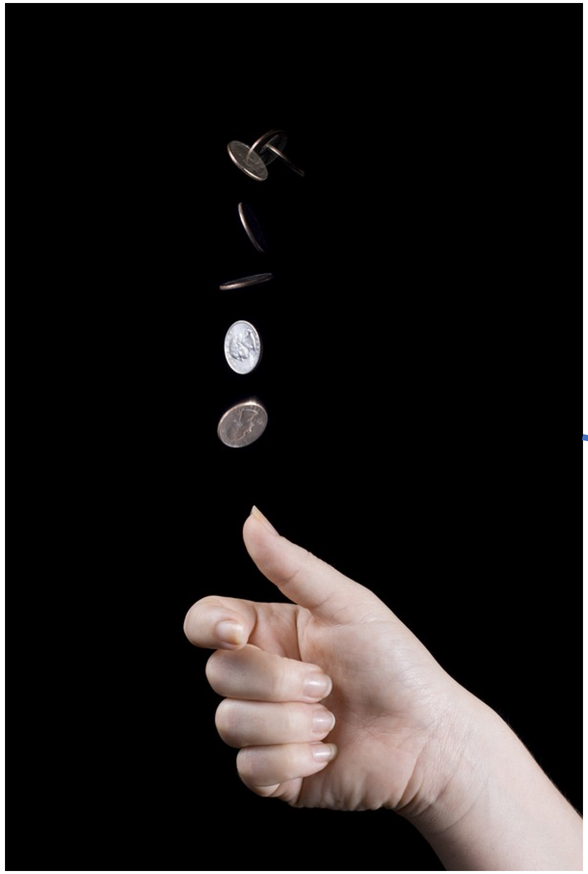
Standard version: $$M_h(y_0) = \left[\frac{1}{n_x}\sum_{i=1}^{n_x} x_i\right] - y_0$$

Weighted version: $$M_h(y_0) = \left[\frac{\sum_{i=1}^{n_x} w_i(y_0)x_i}{\sum_{i=1}^{n_x} w_i(y_0)}\right] - y_0$$

n_x : number of points in the kernel
y_0 : initial mean location
x_i : data points
h: kernel radius

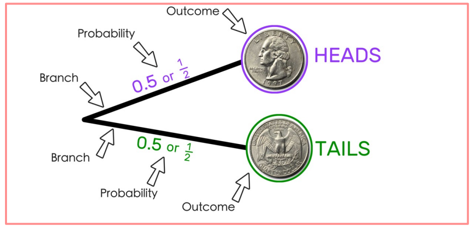
Weights are determined using kernel (masks): uniform, gaussian, epanechnikov

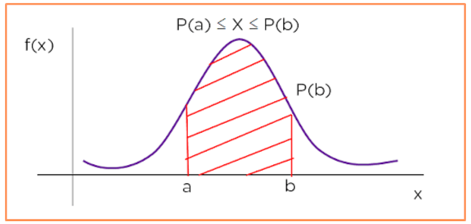
Probability Vs Probability Density Function



Coin Flipping

What is the probability?





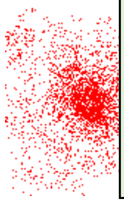
Probability Density Function

Throwing dart

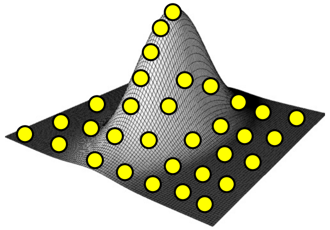
What is Mean Shift

A tool for:
Finding modes (peaks) 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
- …

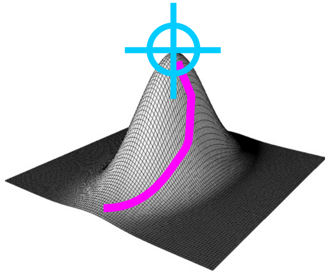


Data



Discrete PDF Representation

Non-parametric
Density GRADIENT Estimation
(Mean Shift)

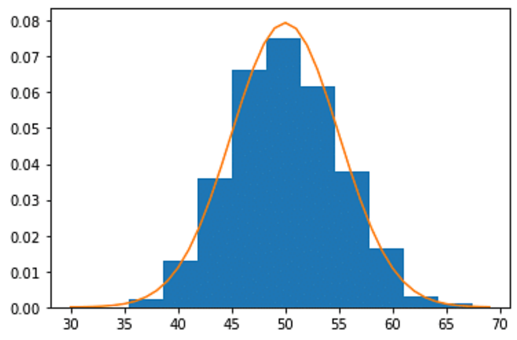


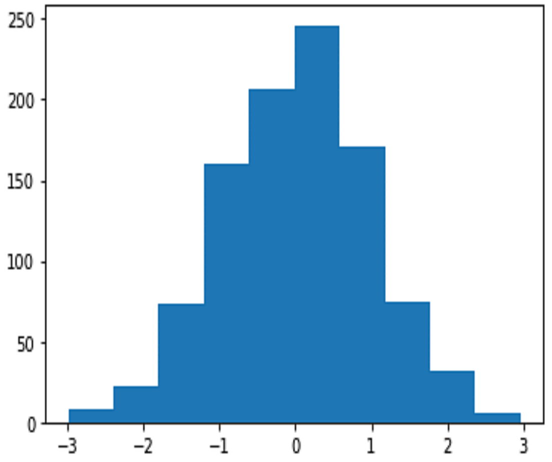
PDF Analysis

Parametric/Non-parametric Estimation

Parametric Density Estimation

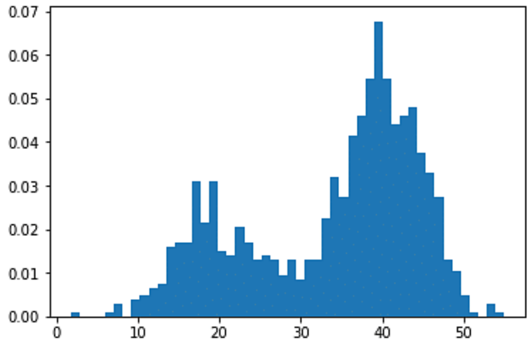
```
# plot the histogram and pdf
pyplot.hist(sample, bins=10, density=True)
pyplot.plot(values, probabilities)
pyplot.show()
```



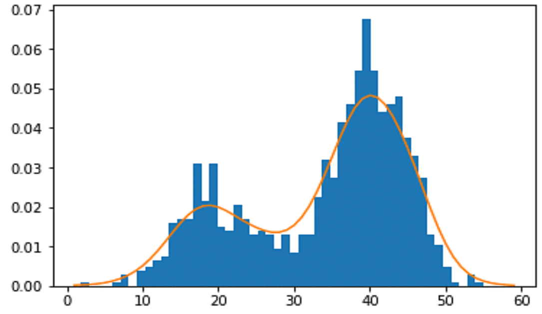


Non-Parametric Density Estimation

```
pyplot.hist(sample, bins=50, density=True)
pyplot.show()
```

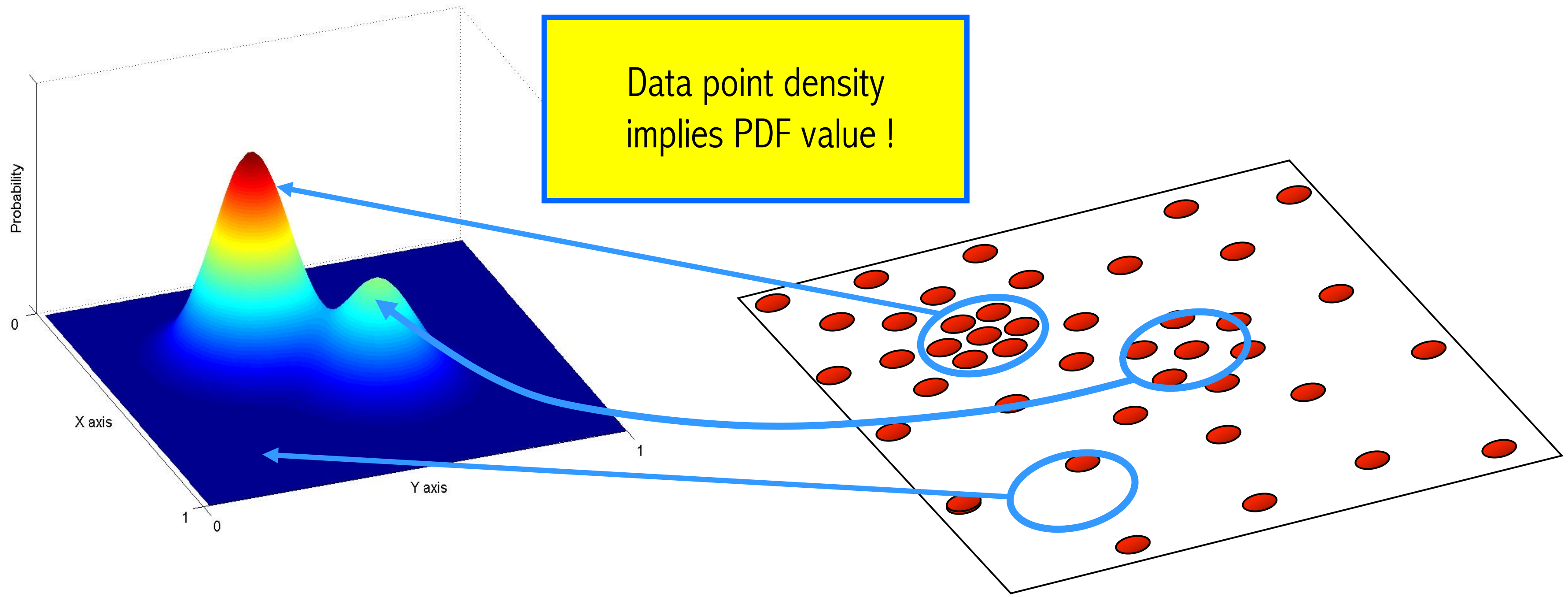



```
# plot the histogram and pdf
pyplot.hist(sample, bins=50, density=True)
pyplot.plot(values[:], probabilities)
pyplot.show()
```



Non-Parametric Density Estimation

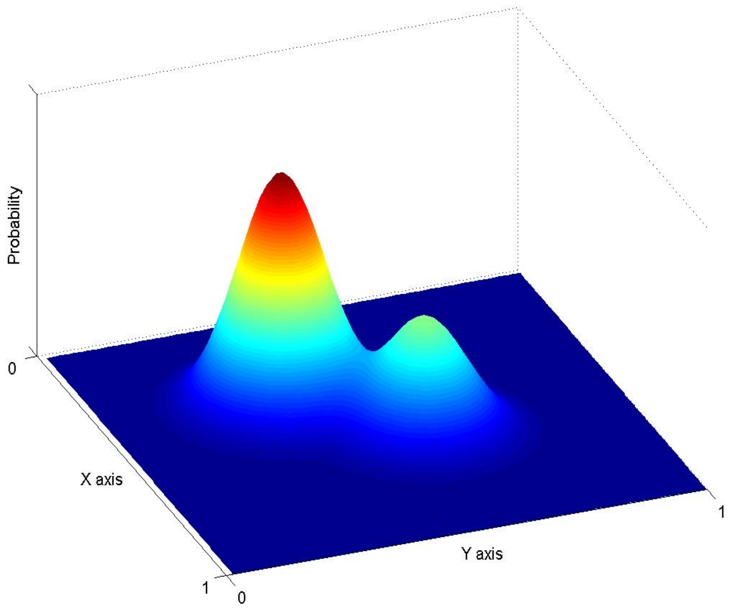
<u>Assumption</u> : The data points are sampled from an underlying PDF



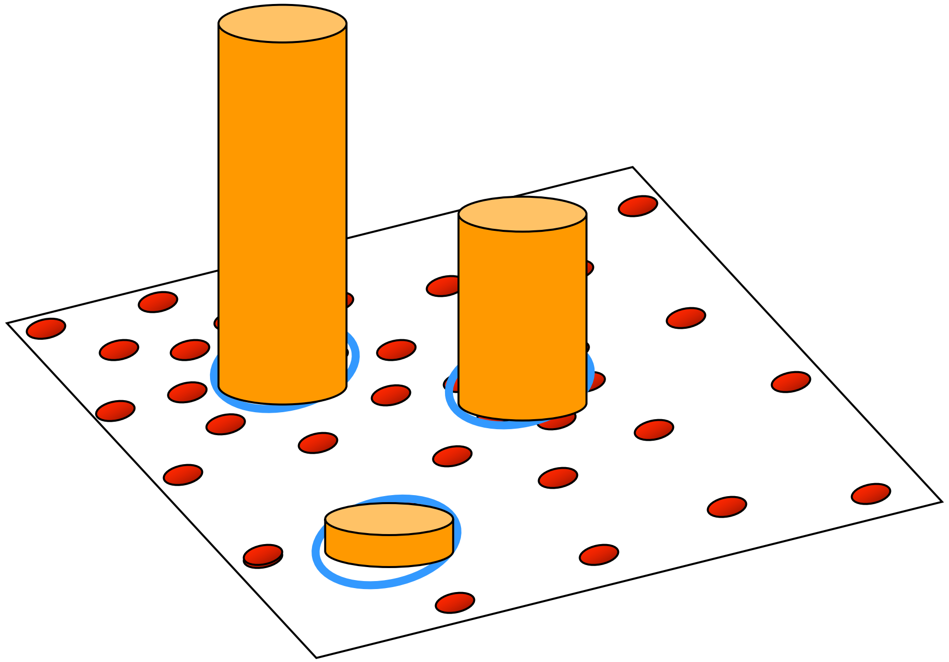
Assumed Underlying PDF

Real Data Samples

Non-Parametric Density Estimation



Assumed Underlying PDF

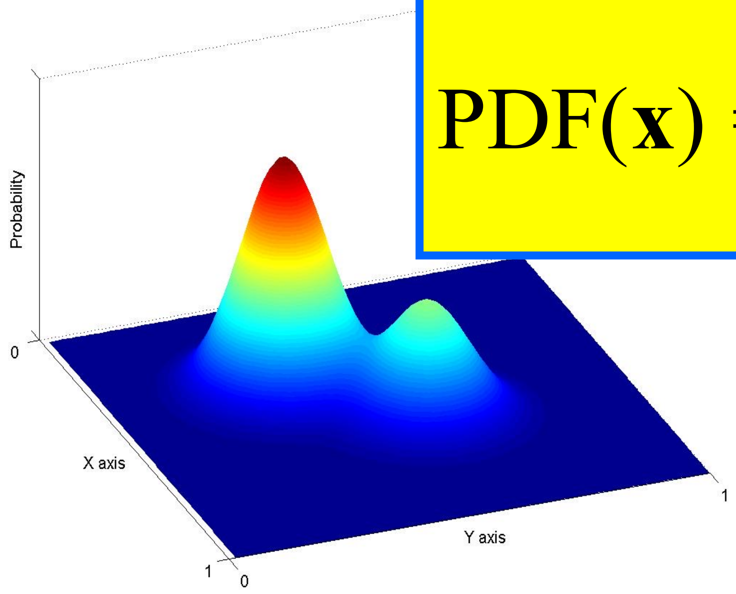


Real Data Samples

Parametric Density Estimation

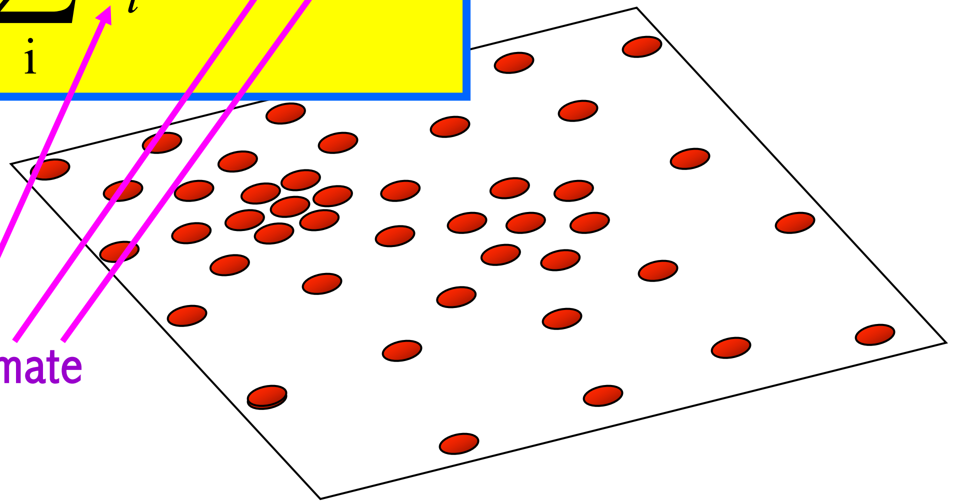
Assumption : The data points are sampled from an underlying PDF

$$\mathrm{PDF}(\mathbf{x}) = \sum_{\mathrm{i}} c_i \cdot e^{-\frac{(\mathbf{x}-\mathbf{u}_i)^2}{2\sigma_i^2}}$$



Assumed Underlying PDF

Estimate

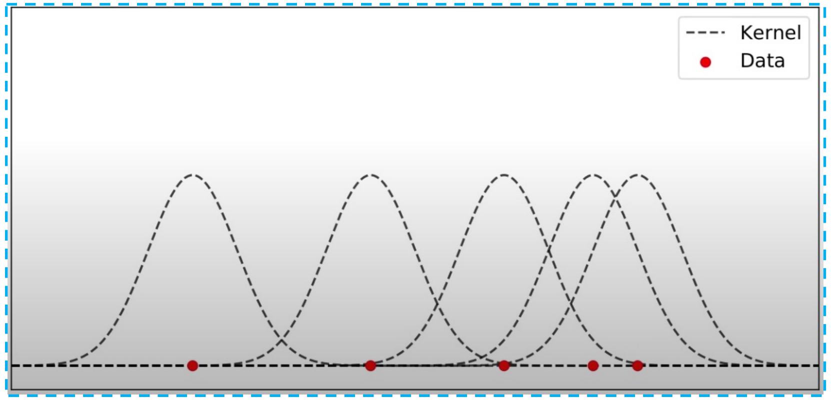


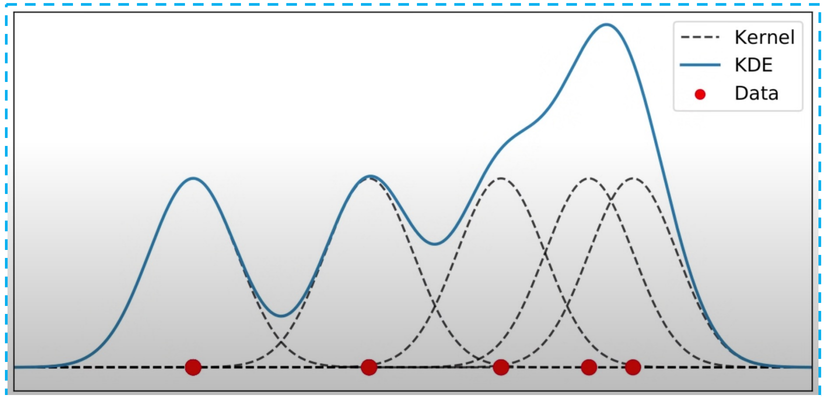
Real Data Samples

Kernel Density Estimation

Kernel Density Estimate:

$$\hat{f}(x) = \frac{1}{N}\sum_{i=1}^{N} K(x - x_i).$$

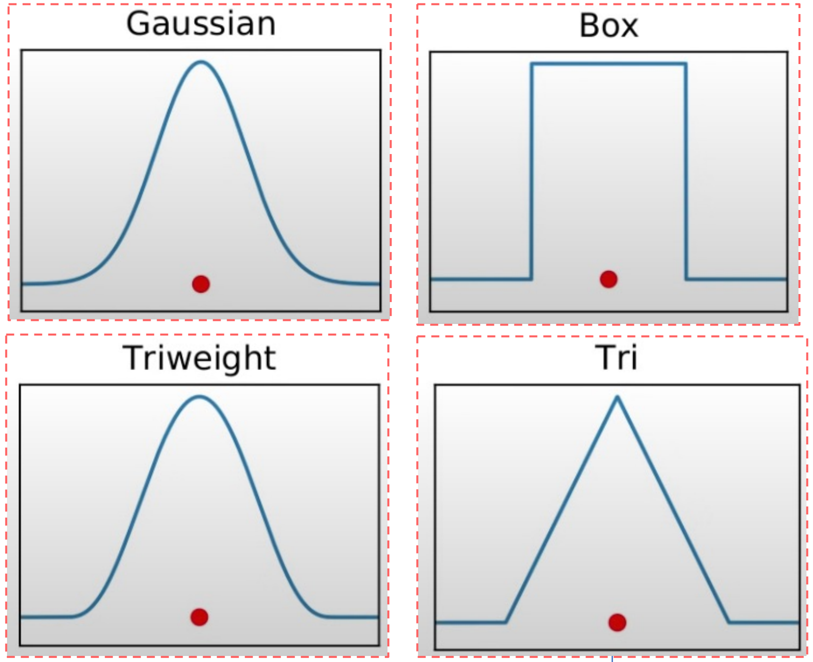




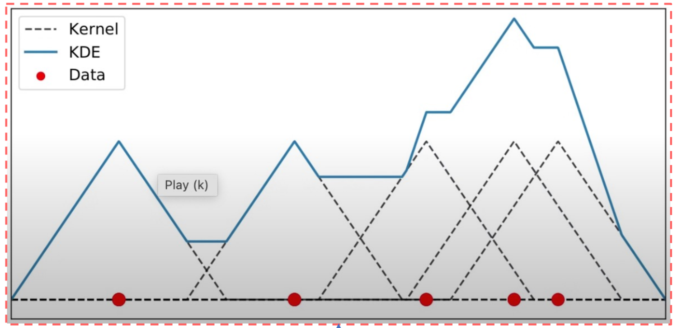
On every data point xi, we place a kernel function K

Kernel is simply a function which satisfies following three properties as mentioned below :

- $K(x) \geq 0$ for every x
- Symmetric: $K(x) = K(-x)$ for every x
- Decreasing: $K'(x) \leq 0$ for every $x > 0$
- Area under the curve of the function must be equal to one



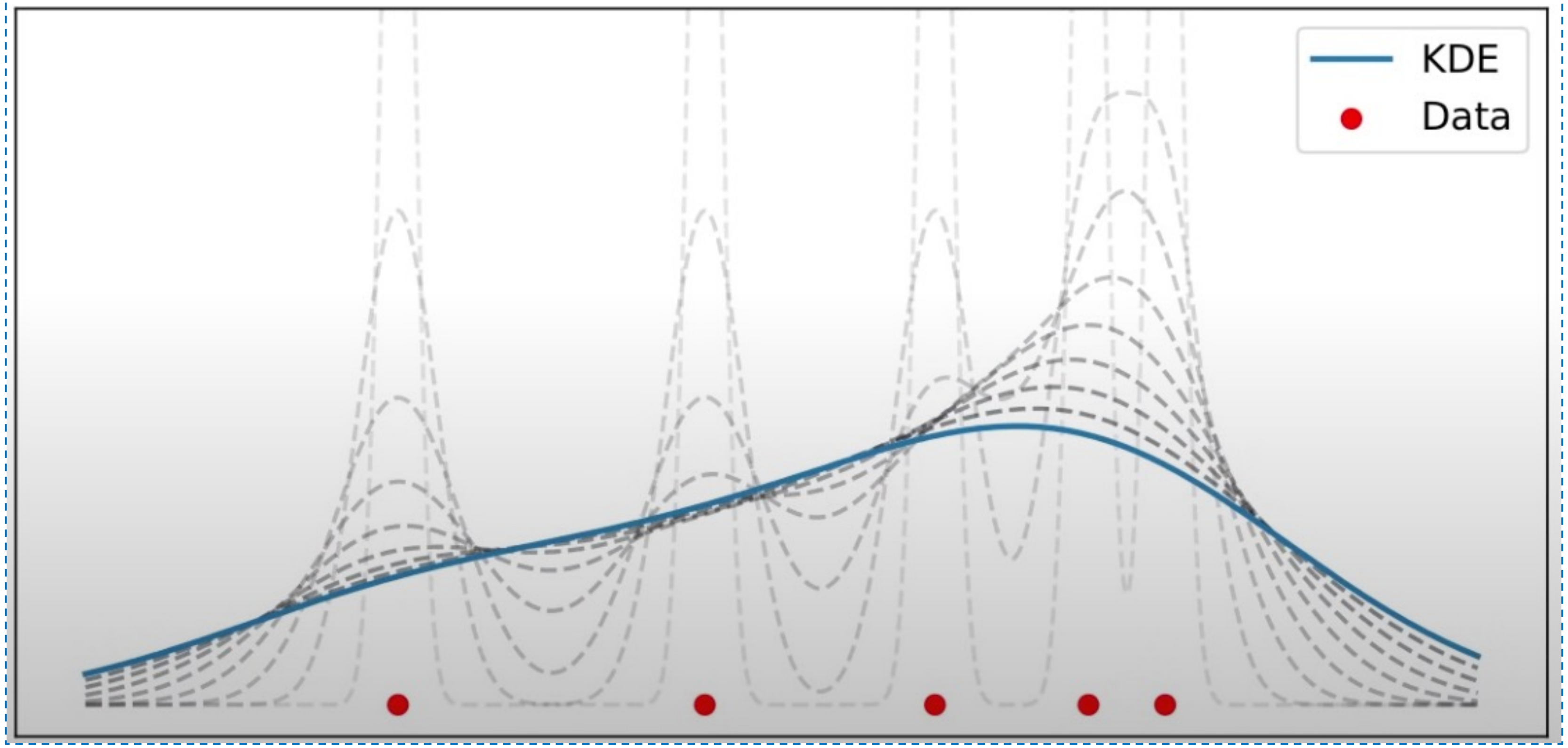
$$f(x) \propto \max(1 - |x|, 0).$$



Kernel Density Estimation (Bandwith)

$$\hat{f}(x) = \frac{1}{Nh}\sum_{i=1}^{N} K\left(\frac{x - x_i}{h}\right)$$

h is used to control bandwith of f(x)



KDE: Example

We have grade obtained by six students in a Computer Vision class.
We are going to construct kernel at each data point using Gaussian kernel function

$X_i = \{65, 75, 67, 79, 81, 91\}$

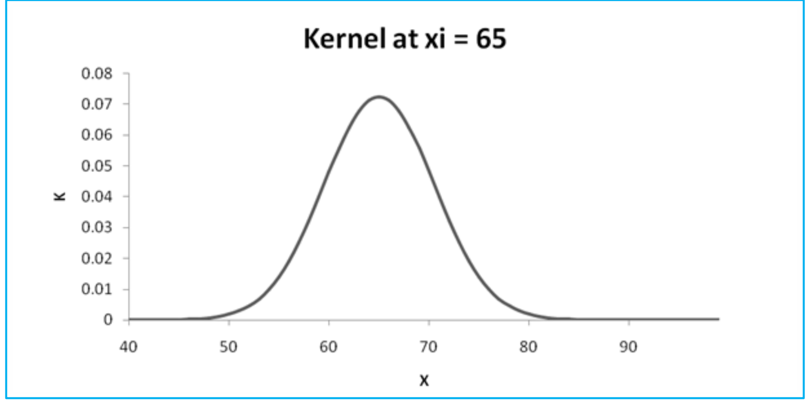
Three inputs are required to develop a kernel curve around a data point:

- The observation data point which is X_i
- The value of h
- A linearly spaced series of data points which houses the observed data points where K values are estimated.

$Xj = \{50,51,52 \ldots . 99\}$

X_j	x_i	h	$A = \frac{1}{h\sqrt{2\pi}}$	$B = -0.5\left(\frac{X_j - x_i}{h}\right)^2$	$K = Ae^B$
50	65	5.5	0.072536	-3.71901	0.00175958
51	**65**	**5.5**	**0.072536**	**-3.23967**	**0.002841733**
52	65	5.5	0.072536	-2.79339	0.00444018
–	–	–	–	–	–
–	–	–	–	–	–
–	–	–	–	–	–
–	–	–	–	–	–
99	**65**	**5.5**	**0.072536**	**-19.1074**	**0.000000000365**
Sum					1.000

Calculation of **K** values for all values of X_j for a given values of X_i and **h** is shown in the table below: where X_i = 65 and **h** = 5.5



KDE: Example

We have grade obtained by six students in a Computer Vision class.
We are going to construct kernel at each data point using Gaussian kernel function

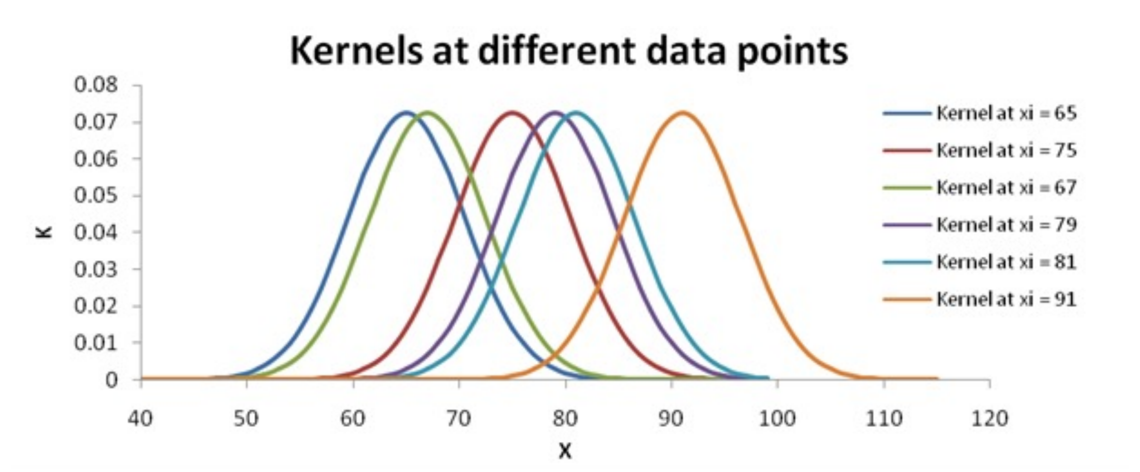
$X_i = \{65, 75, 67, 79, 81, 91\}$

Three inputs are required to develop a kernel curve around a data point:

- The observation data point which is X_i
- The value of h
- A linearly spaced series of data points which houses the observed data points where K values are estimated.

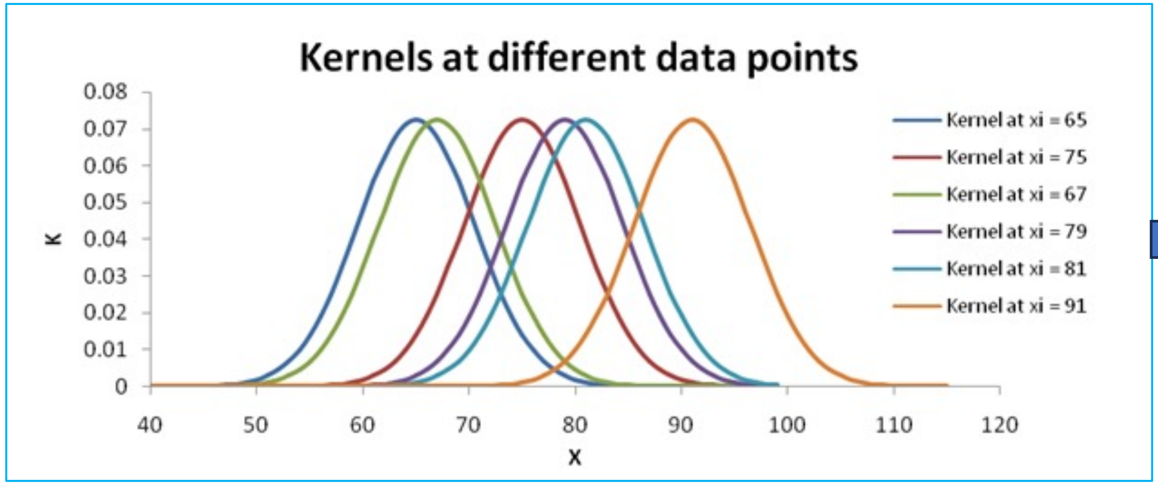
$Xj = \{50,51,52 \dots 99\}$

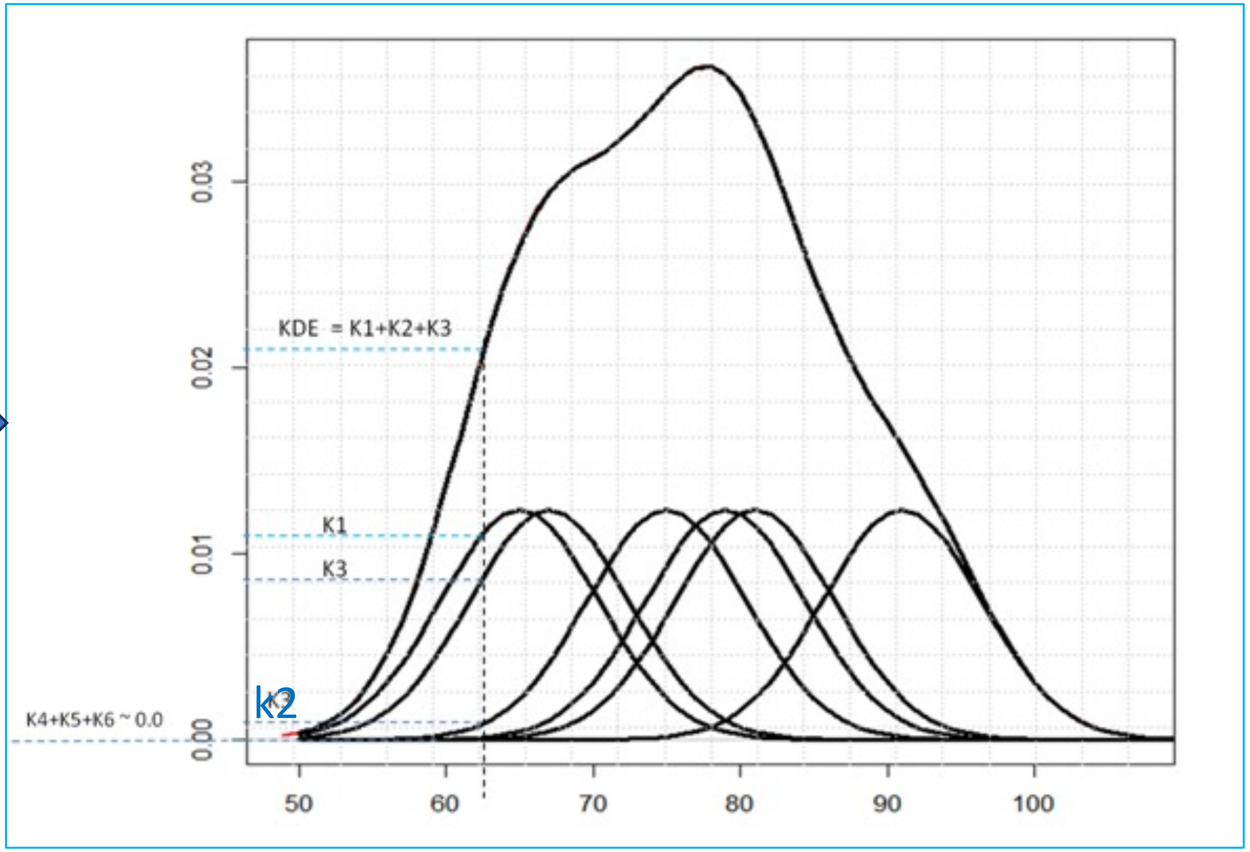
X	K(x) Xi = 65	Xi = 75	Xi=67	Xi=79	Xi=81	Xi=91
50	0.00175958	0.00000237	0.00061093	0.00000007	0.00000001	0.00000000
51	0.00284173	0.00000532	0.00105409	0.00000017	0.00000003	0.00000000
52	0.00444018	0.00001157	0.00175958	0.00000042	0.00000007	0.00000000
53	0.00671214	0.00002433	0.00284173	0.00000102	0.00000017	0.00000000
-	-	-	-	-	-	-
.	.	.	.	.	.	.
-	-	-	-	-	-	-
78	0.00444	0.06251	0.009817	0.071347	0.06251	0.00444
79	0.002842	0.05568	0.006712	0.072536	0.067895	0.006712
80	0.00176	0.047984	0.00444	0.071347	0.071347	0.009817
81	0.001054	0.040007	0.002842	0.067895	0.072536	0.01389
.	.	.	.	.	.	.
99	0.00000	0.0000000	0.00000	0.000000	0.000342567	0.025184586



KDE: Example

$$KDE_j = \frac{1}{n}\sum_{i=1}^{i=n} \frac{1}{h\sqrt{2\pi}} e^{-0.5\left(\frac{x_j - x_i}{h}\right)^2}$$

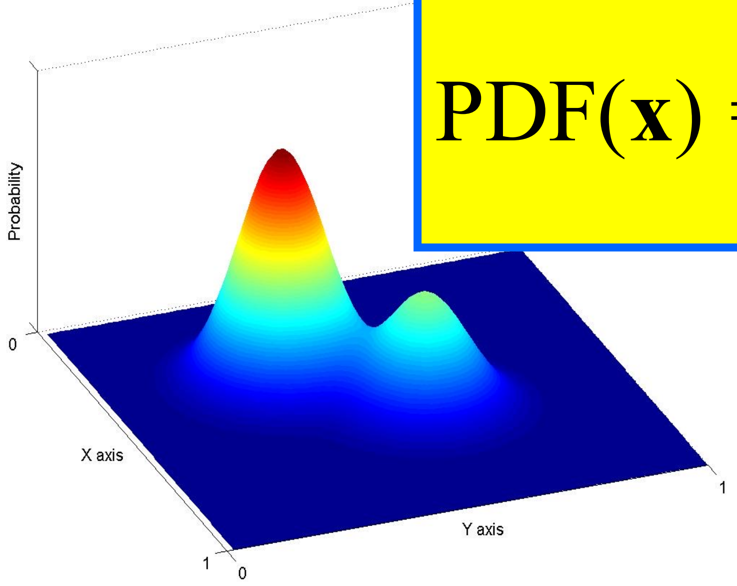




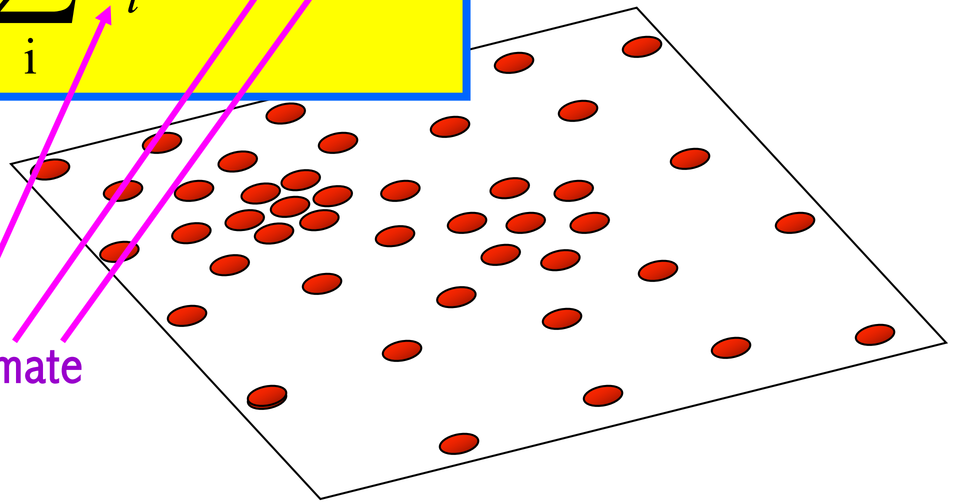
Parametric Density Estimation

Assumption : The data points are sampled from an underlying PDF

$$\mathrm{PDF}(\mathbf{x}) = \sum_{\mathrm{i}} c_i \cdot e^{-\frac{(\mathbf{x}-\mathbf{u}_i)^2}{2\sigma_i^2}}$$



Estimate



Assumed Underlying PDF

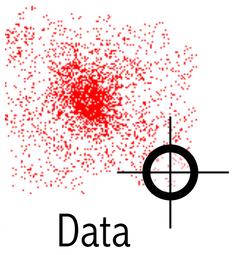
Real Data Samples

Visualization: https://mathisonian.github.io/kde/

Kernel Density Estimation

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

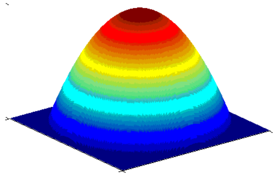
A function of some finite number of data points $x_1 \ldots x_n$



Examples:

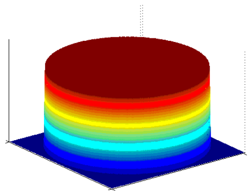
- Epanechnikov Kernel

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



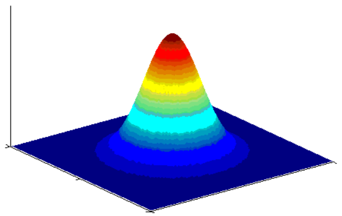
- Uniform Kernel

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



- Normal Kernel

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



Profile and Kernel

Radially Symmetric Kernel

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

$$P(x) = \frac{1}{n}\sum_{i=1}^{n} K(x - x_i) = \frac{1}{n}c\sum_{i=1}^{n} k(\|x - x_i\|^2)$$

$$\nabla P(x) = \frac{1}{n}c\sum_{i=1}^{n} \nabla k(\|x - x_i\|^2)$$

$$\nabla P(x) = \frac{1}{n}2c\sum_{i=1}^{n} (x - x_i)k'(\|x - x_i\|^2)$$

$$\nabla P(x) = \frac{1}{n}2c\sum_{i=1}^{n} (x_i - x)g(\|x - x_i\|^2)$$

$g = -k'(x)$

$$\nabla P(x) = \frac{1}{n}2c\sum_{i=1}^{n} x_i g(\|x - x_i\|^2) - \frac{1}{n}2c\sum_{i=1}^{n} x g(\|x - x_i\|^2)$$

$$\nabla P(x) = \frac{1}{n}2c\sum_{i=1}^{n} g(\|x - x_i\|^2)\left[\frac{\sum_{i=1}^{n} x_i g(\|x - x_i\|^2)}{\sum_{i=1}^{n} g(\|x - x_i\|^2)} - x\right] \Rightarrow \nabla P(x) = \frac{1}{n}2c\sum_{i=1}^{n} g_i\left[\frac{\sum_{i=1}^{n} x_i g_i}{\sum_{i=1}^{n} g_i} - x\right]$$

Kernel Density Estimation

$\underbrace{\text{Estimation}}_{\text{Gradient}}$

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

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 :

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

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

Kernel Density Estimation

Computing The Mean Shift

Gradient

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

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

Computing The Mean Shift

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

Yet another Kernel density estimation !

<u>Simple Mean Shift procedure:</u>

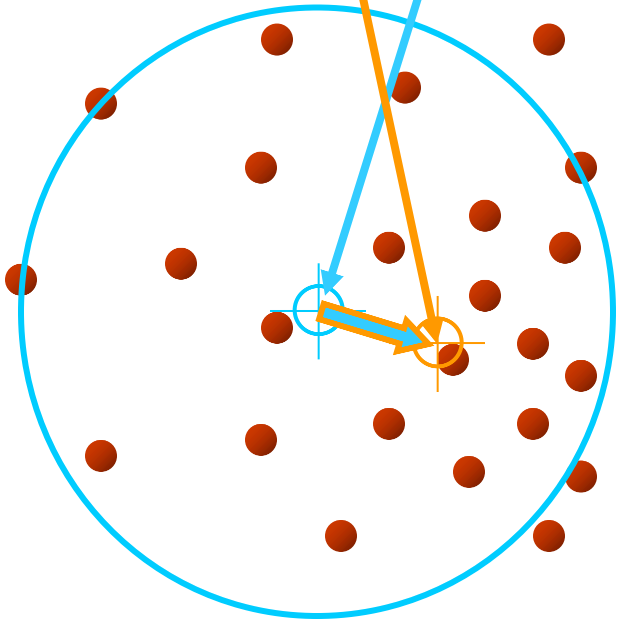
- Compute mean shift vector

Mean Shift is the gradient of the PDF

$$m(x) = \frac{\nabla \mathrm{P(x)}}{\frac{c}{n}\sum_{i=1}^{n} g_i}$$

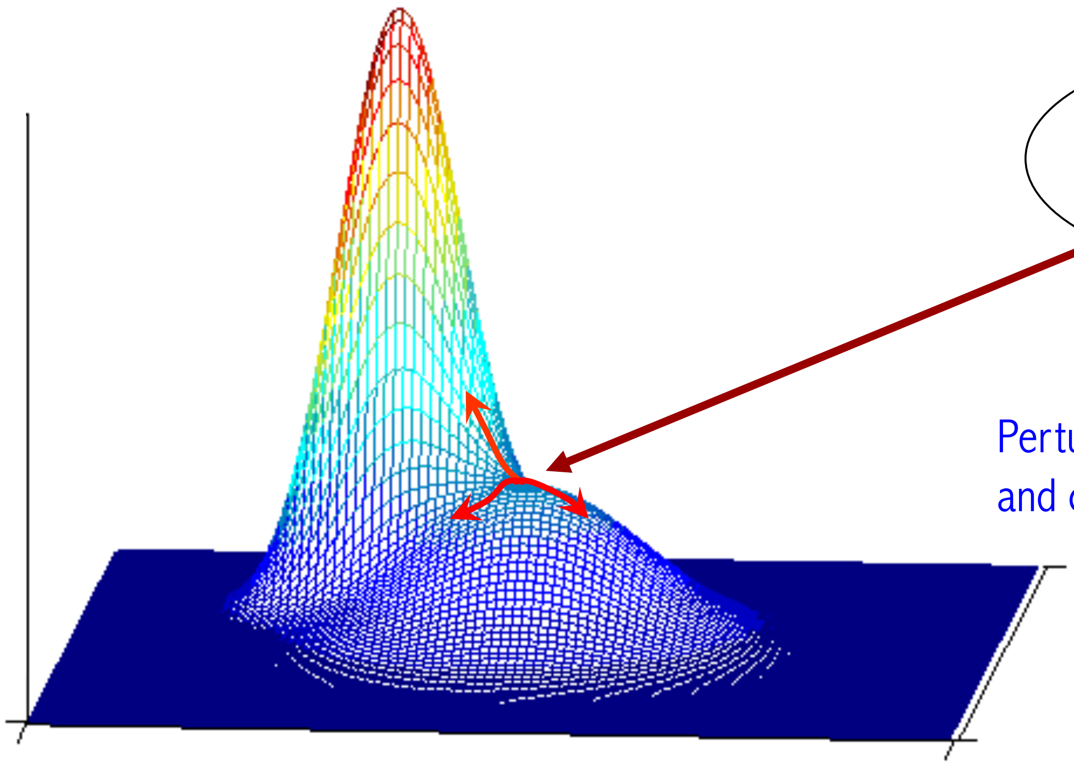
$$\mathbf{m}(\mathbf{x}) = \left[\frac{\sum_{i=1}^{n}\mathbf{x}_i g\left(\frac{\|\mathbf{x}-\mathbf{x}_i\|^2}{h}\right)}{\sum_{i=1}^{n} g\left(\frac{\|\mathbf{x}-\mathbf{x}_i\|^2}{h}\right)} - \mathbf{x}\right]$$

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



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

Mean Shift Mode Detection



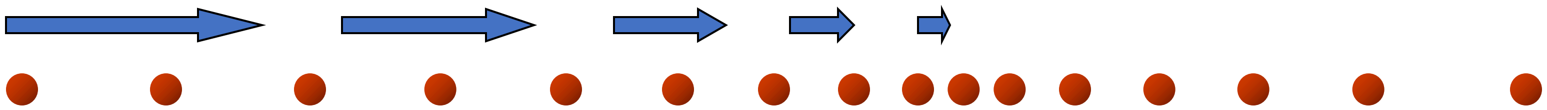
What happens if we reach a saddle point ?

Perturb the mode position and check if we return back

Updated Mean Shift Procedure:

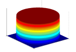
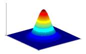
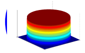
- Find all modes using the Simple Mean Shift Procedure
- Prune modes by perturbing them (find saddle points and plateaus)
- Prune nearby – take highest mode in the window

Mean Shift Properties

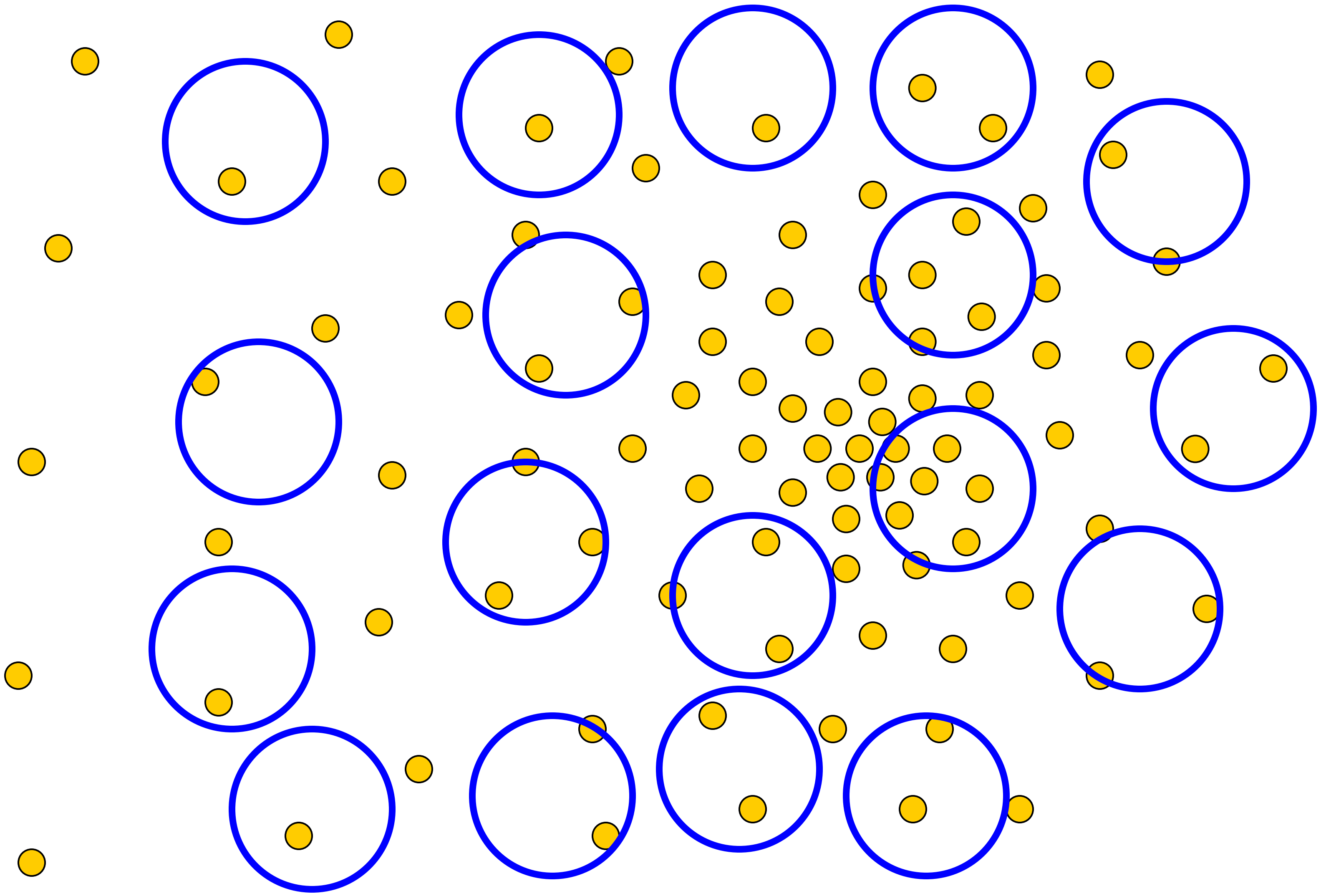


- Automatic convergence speed – the mean shift vector size depends on the gradient itself.
- Near maxima, the steps are small and refined

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 ().

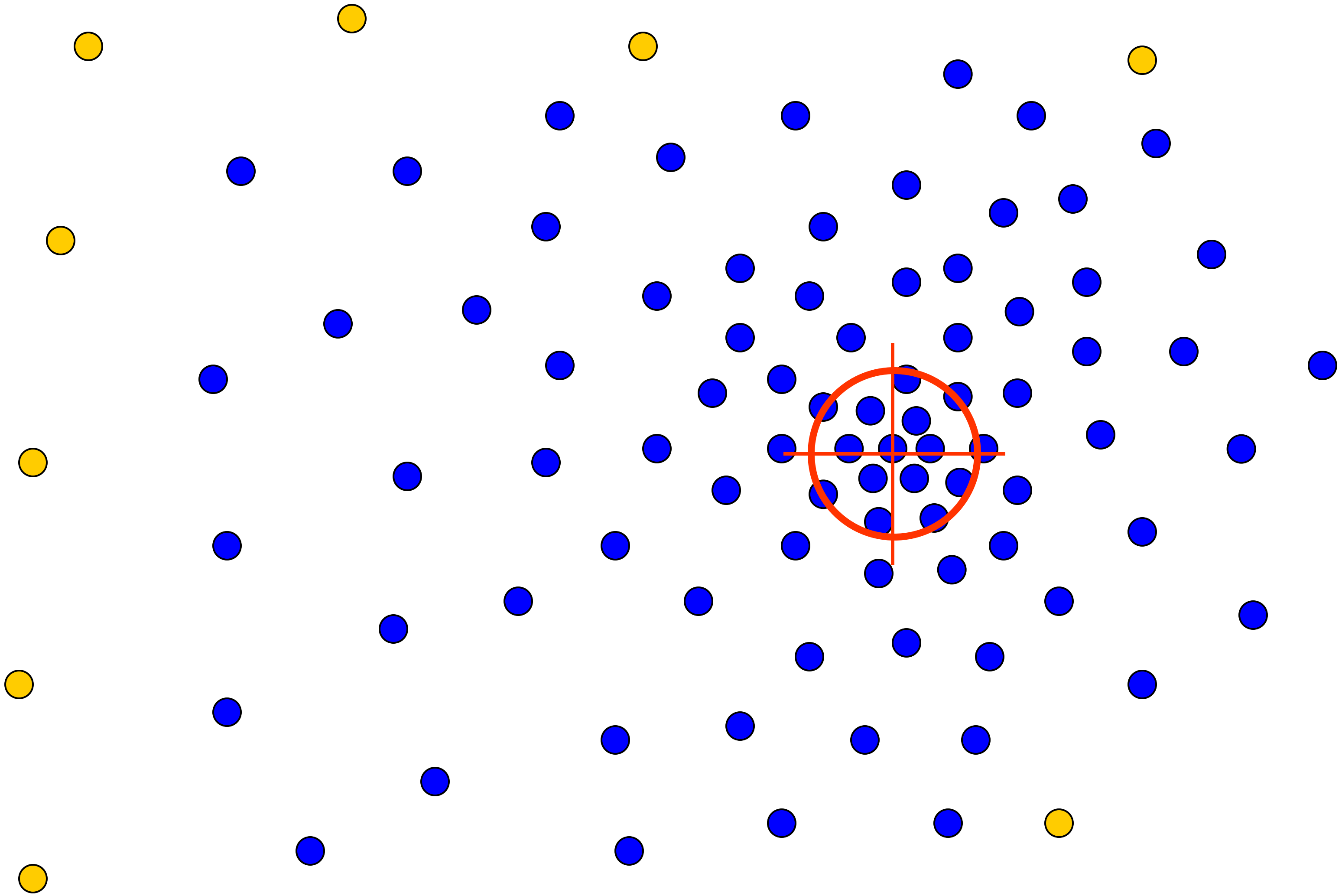
Real Modality Analysis



Tessellate the space with windows

Run the procedure in parallel

Real Modality Analysis



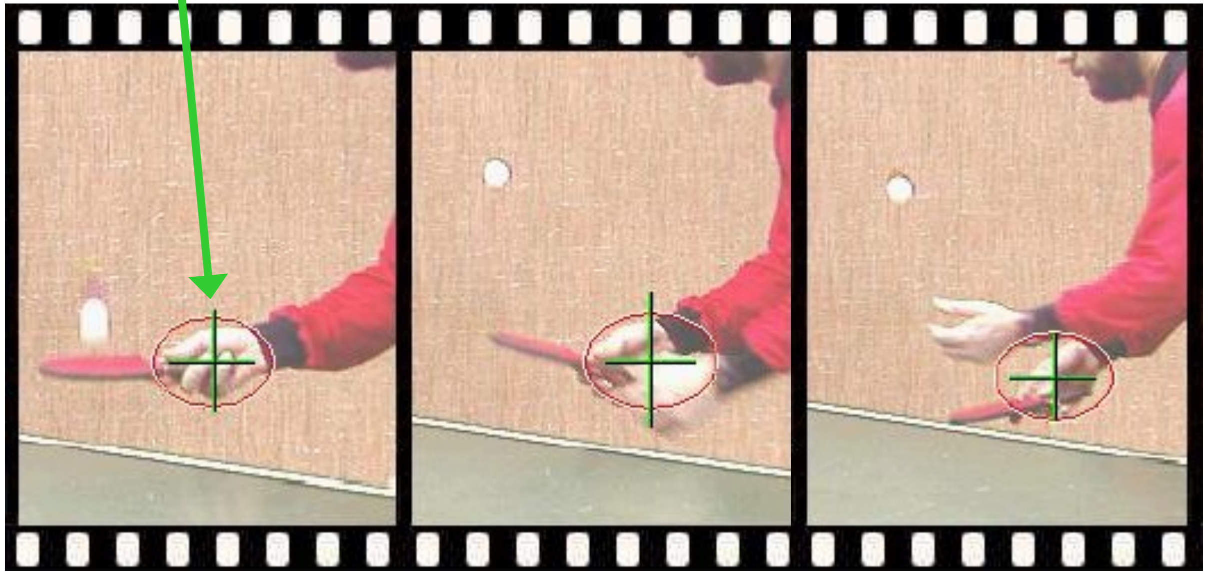
The blue data points were traversed by the windows towards the mode

Outline

- Object Tracking and Its Application
- Object Tracking Based on Template Matching
- Object Tracking Based on Histogram Matching
- Mean Shift Motivation
- Mean Shift Object Tracking
- Object Tracking by SIFT feature
- Face Detection and Tracking

Mean-Shift Object Tracking

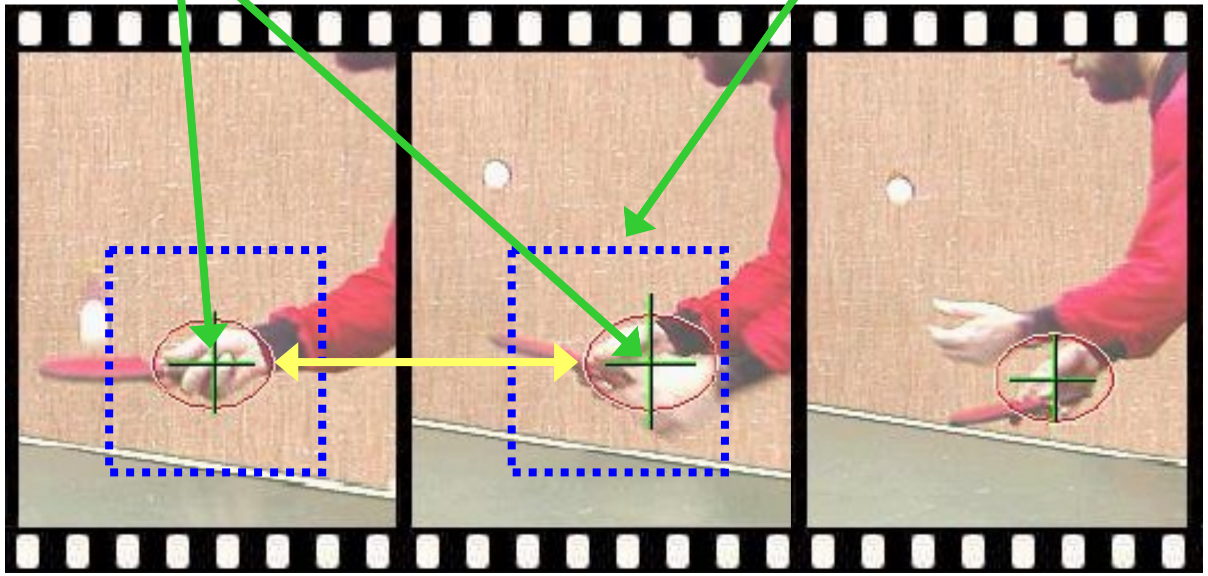
Choose a reference model in the current frame → Choose a feature space → Represent the model in the chosen feature space



… Current frame → …

Mean-Shift Object Tracking

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.



Model

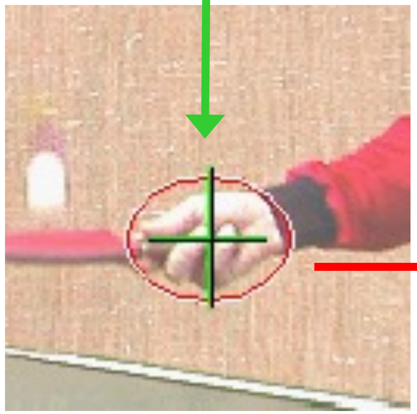
Candidate

… Current frame → …

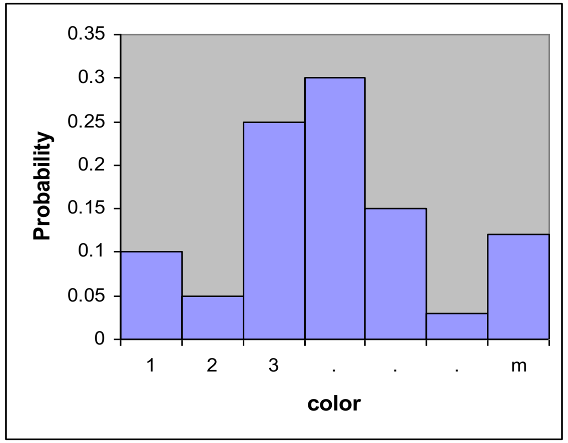
Repeat the same process in the next pair of frames

Mean-Shift Object Tracking

Choose a reference target model → Choose a feature space → Represent the model by its PDF in the feature space



Quantized Color Space



Kernel Based Object Tracking, by Comaniniu, Ramesh, Meer

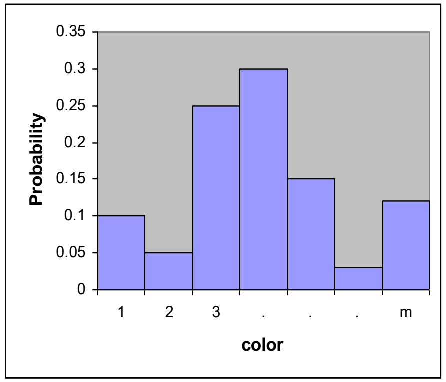
Mean-Shift Object Tracking

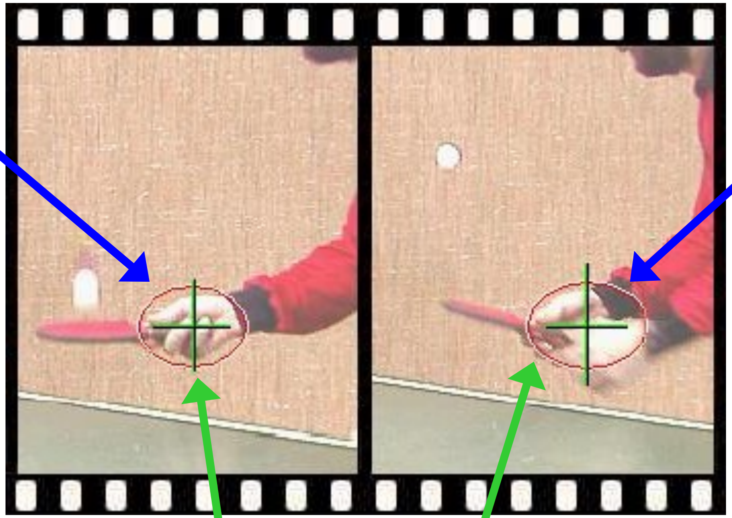
PDF Representation

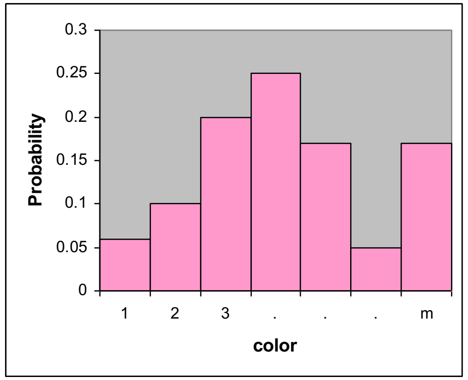
Target Model
(centered at 0)

Target Candidate
(centered at y)

Color distribution by discrete m-bin color histogram







Color distribution by discrete m-bin color histogram

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

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

Similarity Function:

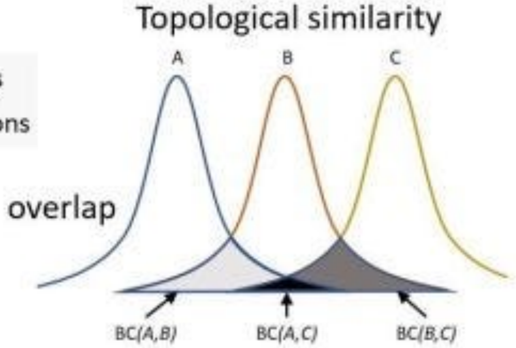
$$f(y) = f\left[\vec{q}, \vec{p}(y)\right]$$

Bhattacharyya Coefficient

Measure the similarity between two probability distributions; It tells us how much overlap there is between the two distributions, and can help us determine how similar or dissimilar they are

Bhattacharyya Distance

Bhattacharyya Dist.(D_B) measures similarity of two discrete or continuous probability distributions
Bhattacharyya Coeff (BC) measures amount of overlap between two statistical samples/populations



$$\boxed{D_B(p,q) = -\ln\big(BC(p,q)\big)}$$

$$BC(p,q) = \sum_{x\in X}\sqrt{p(x)\cdot q(x)} \quad (0 \le BC \le 1)$$

Bhattacharyya coeff for **discrete** Probability Distribution

$$BC(p,q) = \int \sqrt{p(x)\cdot q(x)}\,dx \quad (0 \le BC \le \infty)$$

Bhattacharyya coeff for **continuous** Probability Distribution

D_B between two classes (under normal dist.)
$p_i = N(u_i, \sigma_i^2)$

$$D_B(p,q) = \frac{1}{4}\ln\left(\frac{1}{4}\left(\frac{\sigma_p^2}{\sigma_q^2} + \frac{\sigma_q^2}{\sigma_p^2} + 2\right)\right) + \frac{1}{4}\left(\frac{(\mu_p - \mu_q)^2}{\sigma_p^2 + \sigma_q^2}\right)$$

σ_p^2 – variance of the p^{th} distribution
μ_p – mean of the p^{th} distribution
p, q – two different distributions

D_B between two classes (under multivariate dist.)
$p_i = N(u_i, \Sigma_i)$

$$D_B = \frac{1}{8}(\mu_1 - \mu_2)^T\Sigma^{-1}(\mu_1 - \mu_2) + \frac{1}{2}\ln\left(\frac{\det\Sigma}{\sqrt{\det\Sigma_1\cdot\det\Sigma_2}}\right)$$

μ_i – means of the distributions
Σ_i – covariances of the distributions

$$\Sigma = \frac{\Sigma_1 + \Sigma_2}{2}$$

Mean Shift is applied to find find the solution.

Image credit: Danny Butvinik

Mean-Shift Object Tracking

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

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

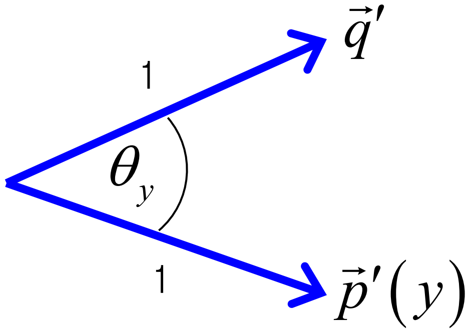
Similarity function: $f(y) = f\left[\vec{p}(y), \vec{q}\right] = ?$

The Bhattacharyya Coefficient

Large f means good match between candidate and target model

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

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

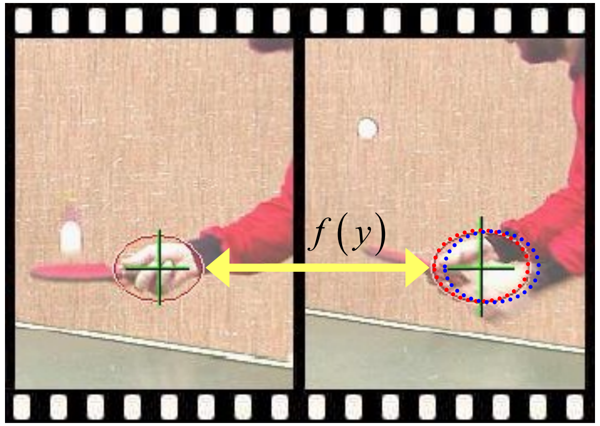


$$\cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|}$$

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

Mean-Shift Object Tracking

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 →

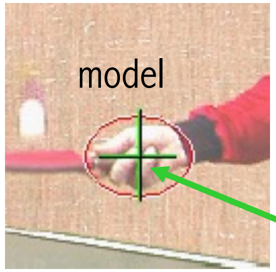
f is not smooth → Gradient-based optimizations are not robust

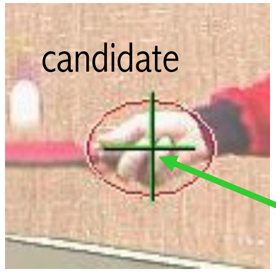
Solution:

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

Mean-Shift Object Tracking: Distribution

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





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

- Peripheral pixels are affected by occlusion and background interference

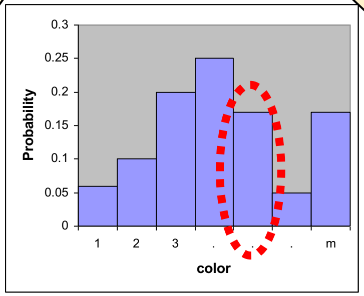
$b(x)$ The color bin index (1..*m*) of pixel *x*

Probability of feature u in model

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

Normalization factor

Pixel weight

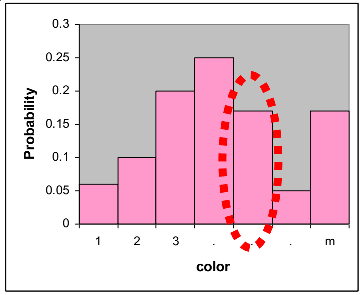


Probability of feature u in candidate

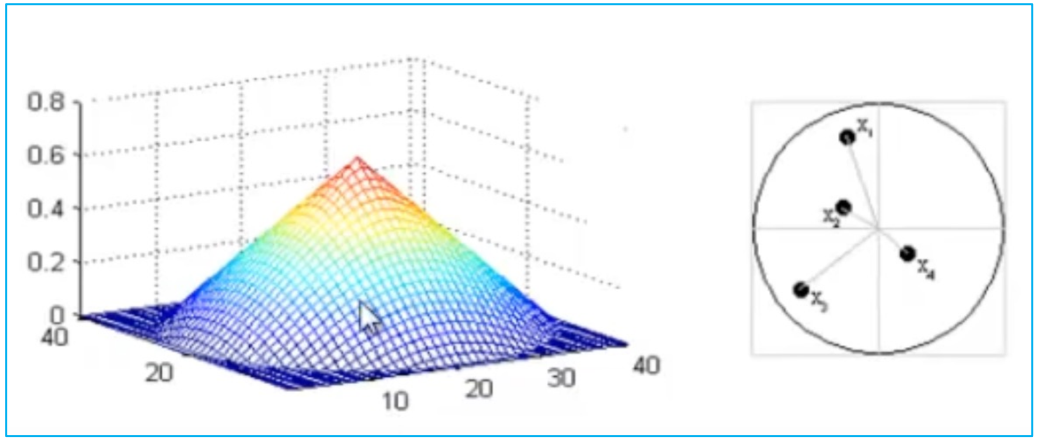
$$p_u(y) = C_h \sum_{b(x_i)=u} \underbrace{k\left(\left\|\frac{y-x_i}{h}\right\|^2\right)}$$

Normalization factor

Pixel weight



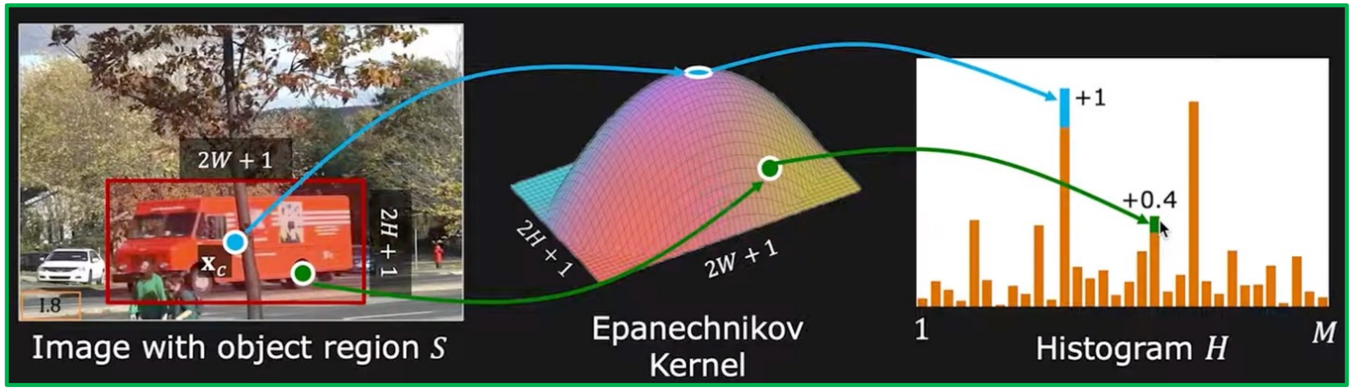
How to estimate Distribution

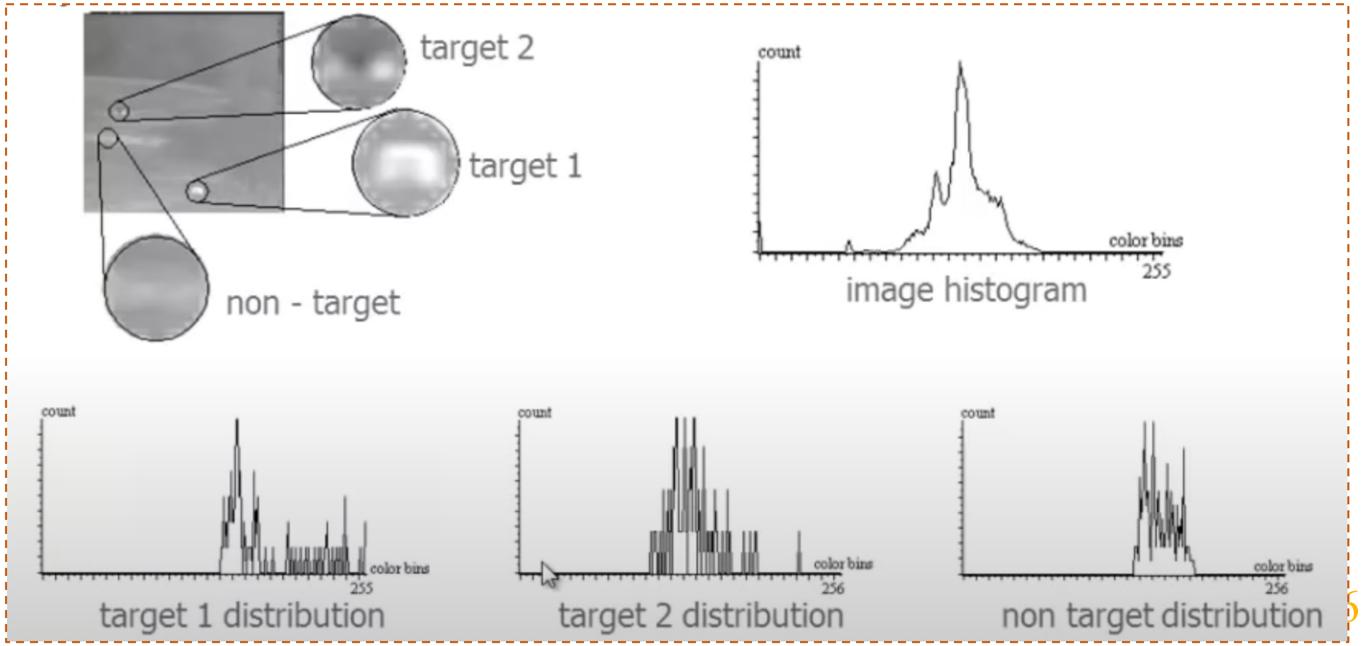


S(x_i) is the color at x_i

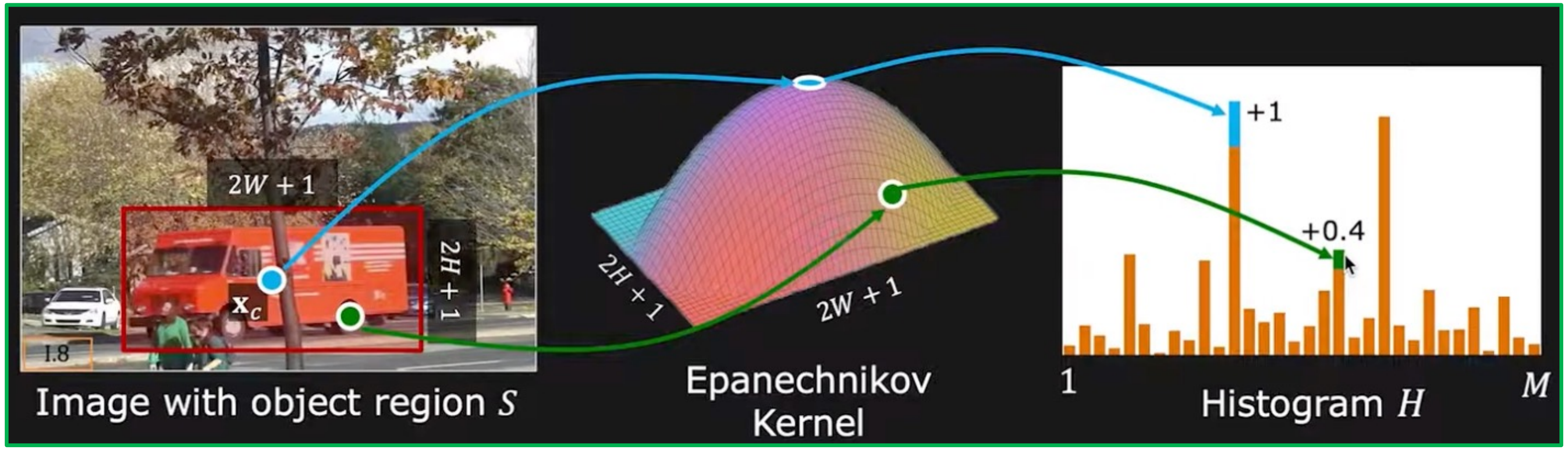
$$p(u) = C \sum_{\mathbf{x}_i \in S} k\left(\|\mathbf{x}_i\|^2\right) \delta[\, S(\mathbf{x}_i) - u]$$

$$\delta(S(x_i) - u) = \begin{cases} 1 \text{ if } S(x_i) = u \\ 0 \text{ if } S(x_i) \neq u \end{cases}$$





Weighted Histogram



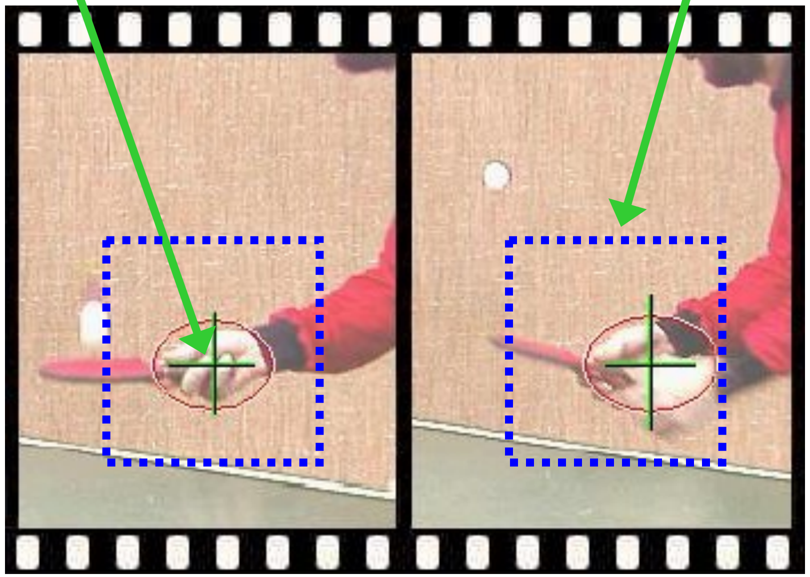
Weighted histogram gives more importance to pixels at center

Epanechnikov Kernel:

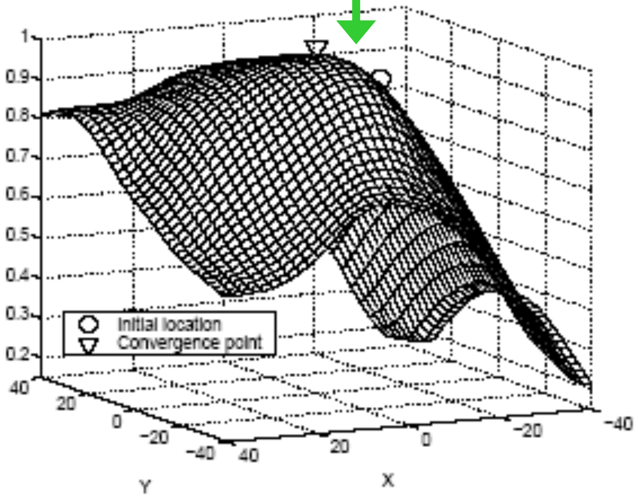
$$k(\tilde{\mathbf{x}}) = \begin{cases} 1 - \|\tilde{\mathbf{x}}\|^2, & \|\tilde{\mathbf{x}}\| < 1 \\ 0, & \text{otherwise} \end{cases} \qquad \tilde{\mathbf{x}} = \begin{bmatrix} (x - x_c)/W \\ (y - y_c)/H \end{bmatrix}$$

Mean-Shift Object Tracking

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.



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



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

Mean-Shift Object Tracking

$$f(y) = \sum_{u=1}^{m} \sqrt{p_u(y) q_u}$$

Model location: y_0

Candidate location: y

Linear approx. (around y_0)

$$f(y) \approx \underbrace{\frac{1}{2}\sum_{u=1}^{m}\sqrt{p_u(y_0) q_u}}_{\text{Independent of } y} + \underbrace{\frac{1}{2}\sum_{u=1}^{m} \boxed{p_u(y)} \boxed{\sqrt{\frac{q_u}{p_u(y_0)}}}}$$

Independent of y

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

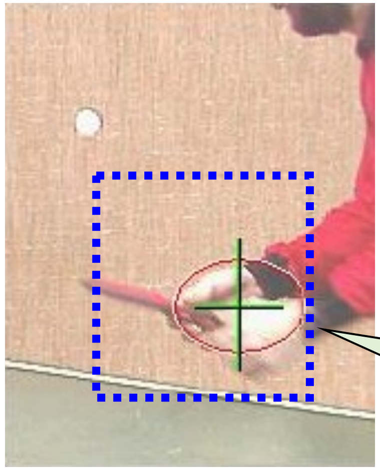
$$\frac{C_h}{2} \sum_{i=1}^{n} \boxed{w_i} \boxed{k\left(\left\|\frac{y - x_i}{h}\right\|^2\right)}$$

Density estimate!
(as a function of y)

Mean-Shift Object Tracking

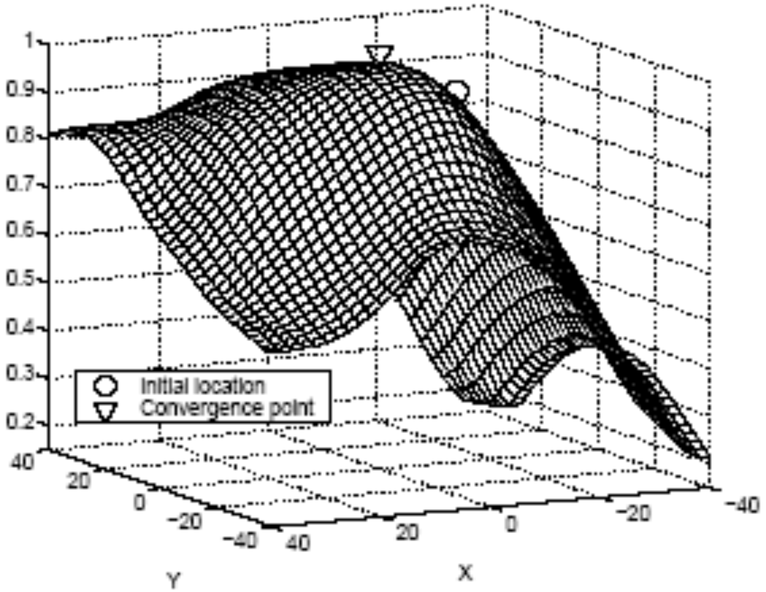
The mode of $$\frac{C_h}{2}\sum_{i=1}^{n} w_i k\left(\left\|\frac{y-x_i}{h}\right\|^2\right)$$ = sought maximum

Important Assumption:



The target representation provides sufficient discrimination

One mode in the searched neighborhood



Mean-Shift Object Tracking

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

Mean-Shift Object Tracking

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:

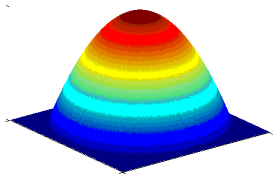
Find mode of $c\sum_{i=1}^{n} w_i k\left(\left\|\frac{y - x_i}{h}\right\|^2\right)$ using $y_1 = \frac{\sum_{i=1}^{n} x_i w_i g\left(\left\|\frac{y_0 - x_i}{h}\right\|^2\right)}{\sum_{i=1}^{n} w_i g\left(\left\|\frac{y_0 - x_i}{h}\right\|^2\right)}$

Mean-Shift Object Tracking

A special class of radially symmetric kernels:

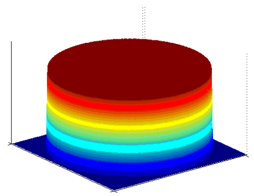
$$K(x) = ck\left(\|x\|^2\right)$$

Epanechnikov kernel:



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

Uniform kernel:



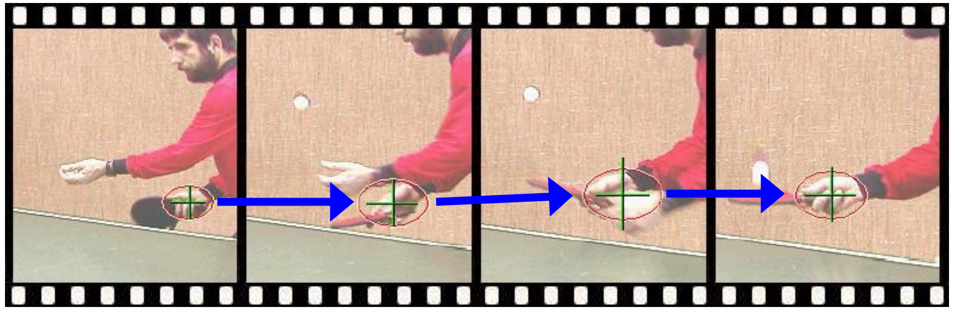
$$g(x) = -k(x) = \begin{cases} 1 & \text{if } \|x\| \le 1 \\ 0 & \text{otherwise} \end{cases}$$

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

Mean-Shift Object Tracking

Problem:

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



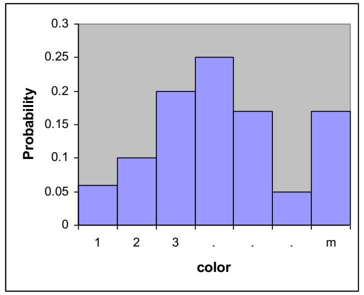
Solution:

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

Mean Shift Summary

1 Calculate distribution (q)

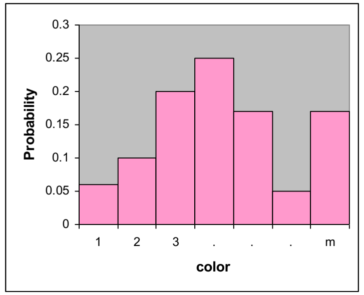
$$q_u = C \sum_{b(x_i)=u} k\left(\left\|x_i\right\|^2\right)$$



Initialize estimated center (y1 = y0)

2 Calculate distribution (p)

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



3 Calculate weight (w)

$$\sqrt{\frac{q_u}{p_u(y_0)}}$$

4 Estimate new target center (y1)

$$y_1 = \frac{\sum_{i=1}^{n} x_i w_i}{\sum_{i=1}^{n} w_i}$$

5 d < α

6′ False

6 True

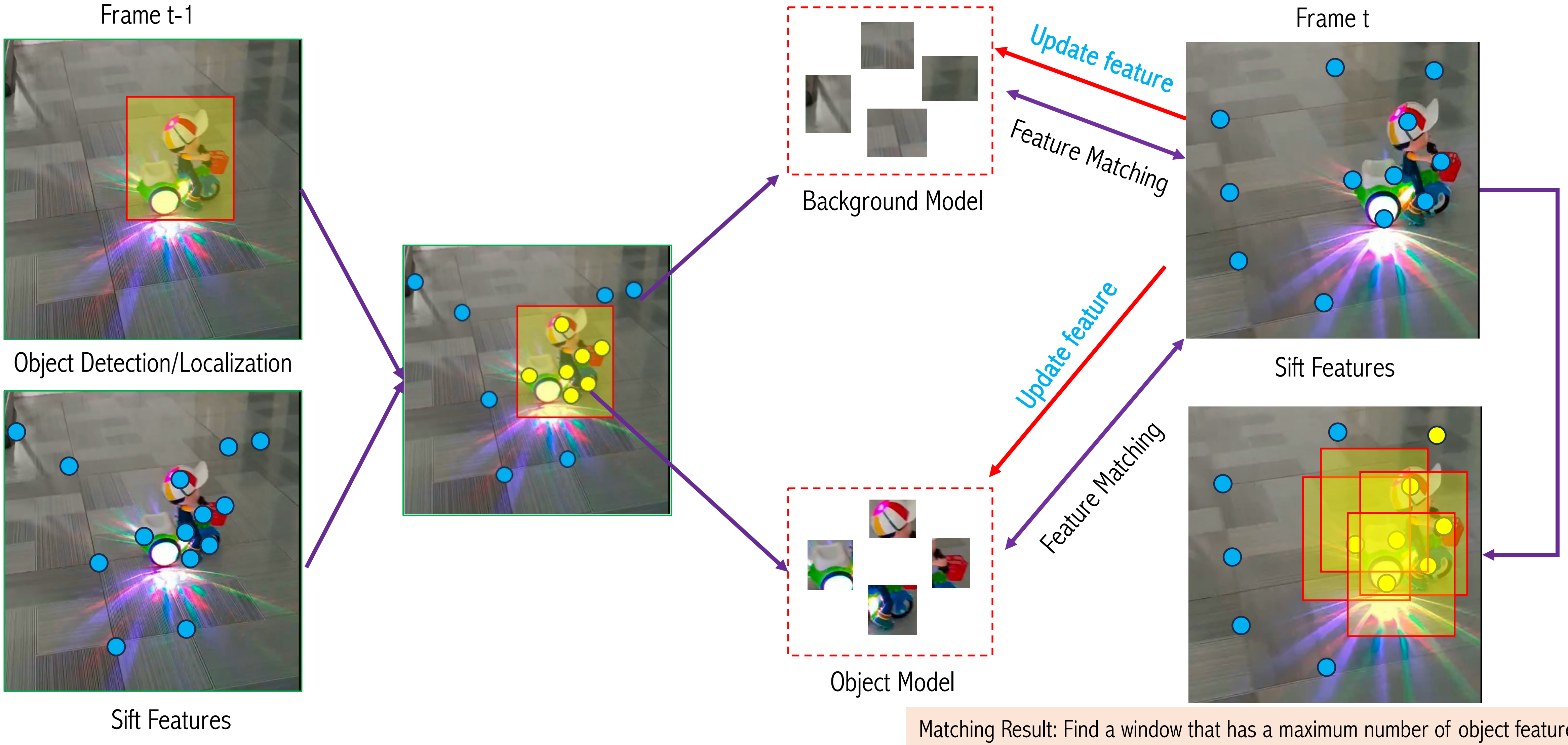
Update target center (y0 = y1)

7 Repeat until end of the sequence

9

Outline

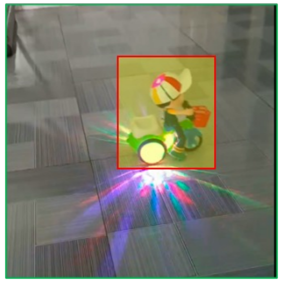
Tracking By Feature Detetection



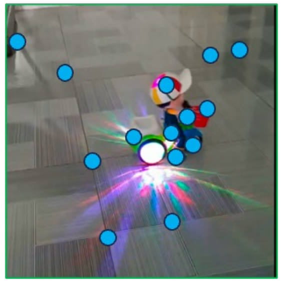
Matching Result: Find a window that has a maximum number of object features

Tracking By Feature Detetection

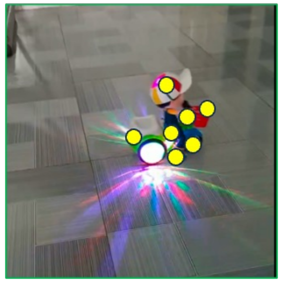
Frame 1



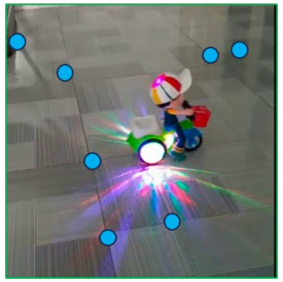
1. Select a bounding box W1 as object/target



2. Compute Sift Feature

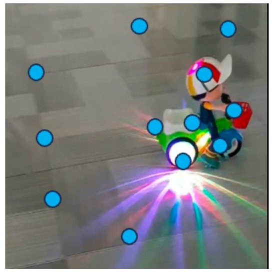


3. Classify features within the box W1 as object and asign them to set 01



4. Classify remaining features as background and asign them to set 01

Frame t



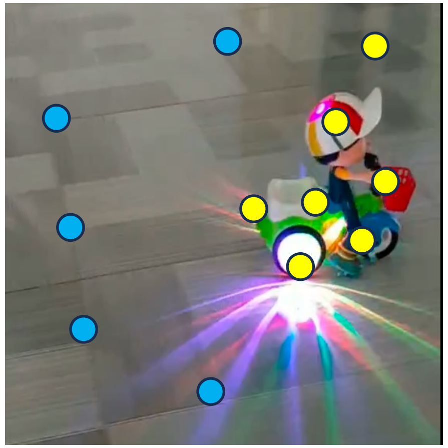
1. Compute Sift features. Sift descriptors $\{v_1, v_2, \ldots, v_k\}$

2. For each Sift descriptor v_i:

a) Compute distance d_O between v_i and the best match in object set O_{t-1}

b) Compute distance d_B between v_i and the best match in background set B_{t-1}

c) $$\mathcal{L}(v_i) = \begin{cases} +1 \ if \ \frac{d_O}{d_B} < 0.5 \ (object) \\ -1 \ if \ \frac{d_O}{d_B} < 0.5 \ (background) \end{cases}$$

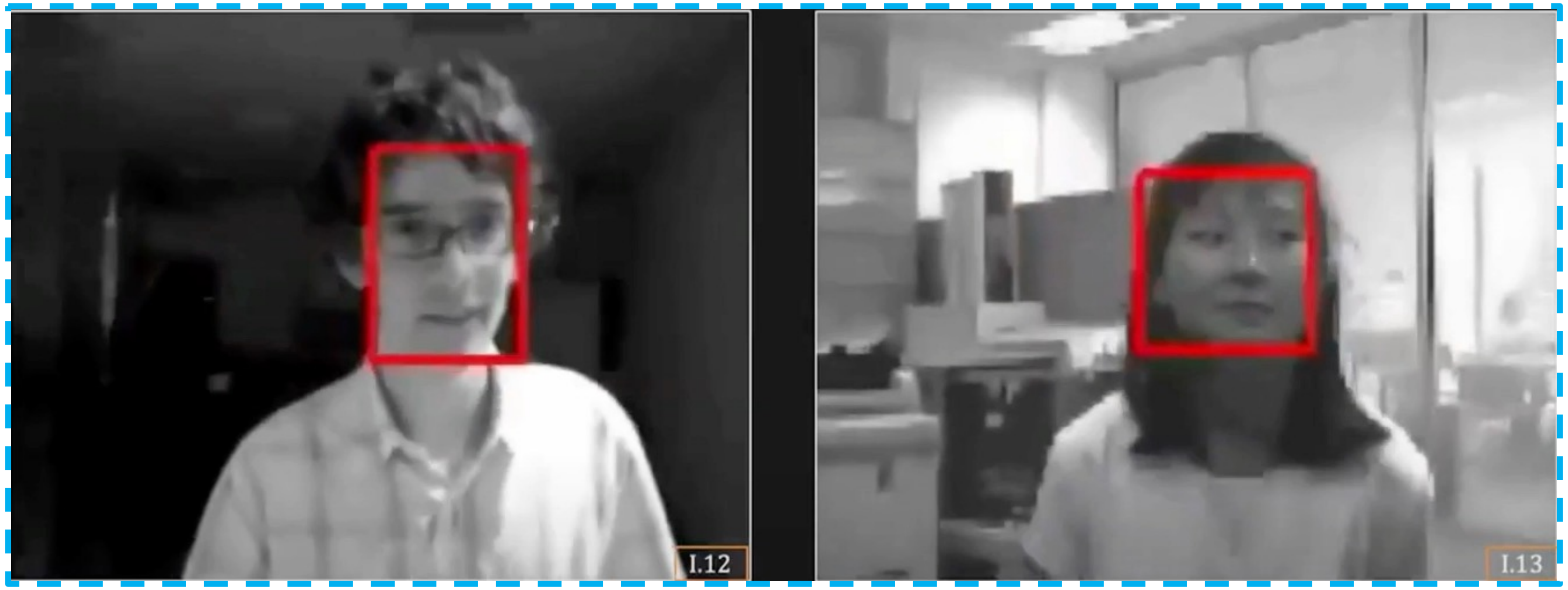


-1 +1

Discusion: How to find object location in frame t

Tracking Results

Robust to scale and orientation

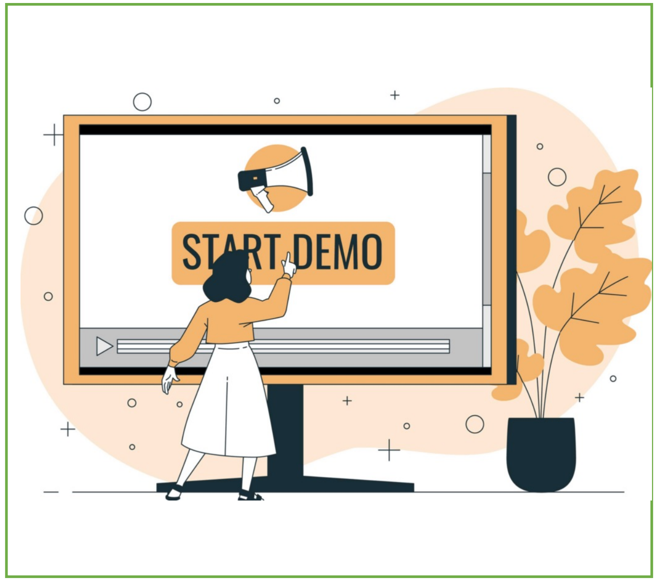


Credit video: Professor Shree Nayar who is faculty in the Computer Science Department, School of Engineering and Applied Sciences, Columbia University

Outline

- Object Tracking and Its Application
- Object Tracking Based on Template Matching
- Object Tracking Based on Histogram Matching
- Mean Shift Motivation
- Mean Shift Object Tracking
- Tracking by Feature Detection
- Face Detection and Tracking

Face Detection and Tracking



Face detection without tracking

Face detection with tracking-based template

Face detection with tracking-based meanshift

Credits slides and videos: Several slides and video are borrowed from *Professor Shree Nayar who is faculty in the Computer Science Department, School of Engineering and Applied Sciences, Columbia University*