Technischen Universität München
Winter Semester 2012/2013

TRACKING and DETECTION in COMPUTER VISION

Mean Shift Tracking

Slobodan Ilić

Agenda

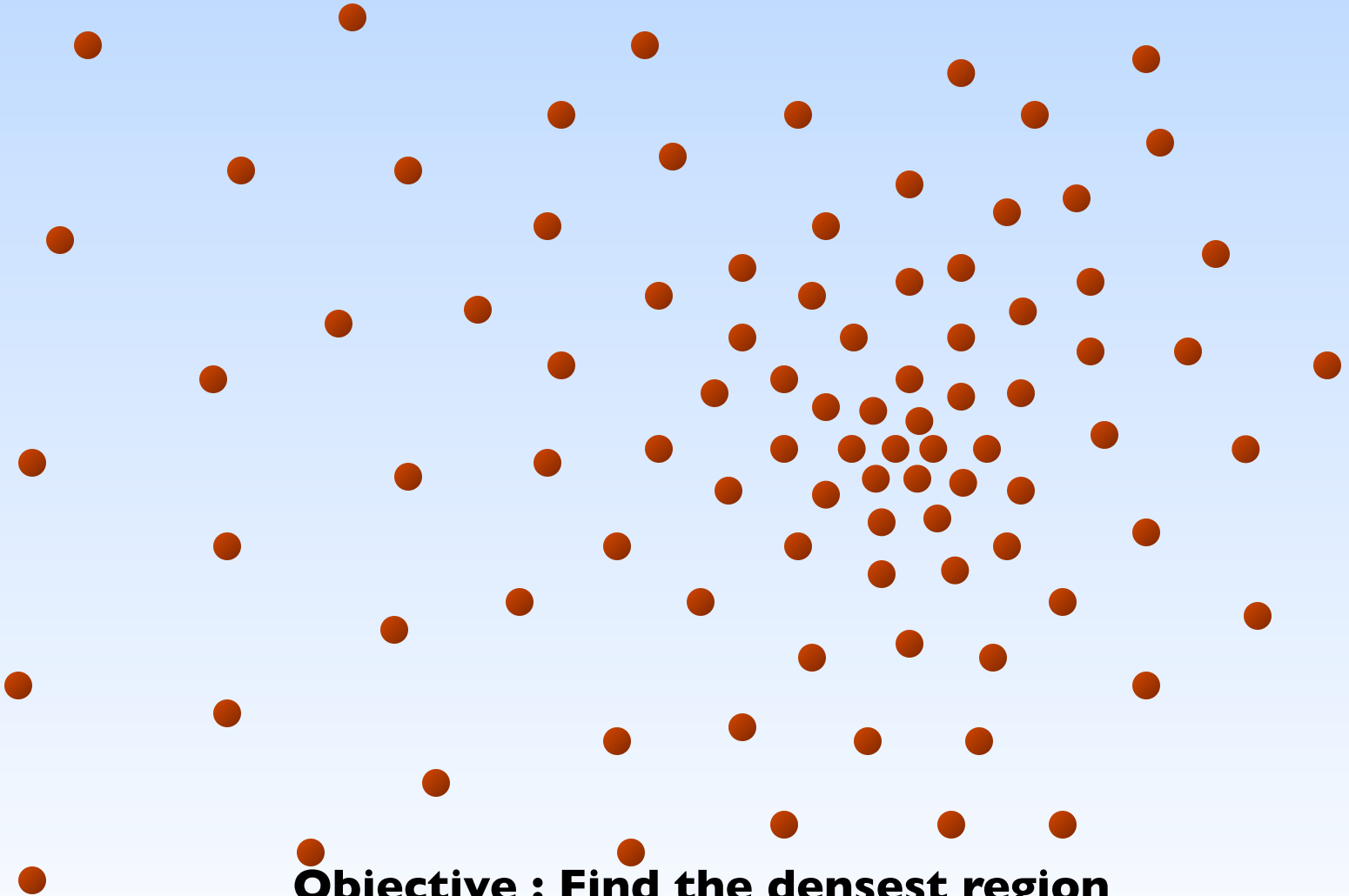
Mean Shift Theory

- What is Mean Shift ?
- Density Estimation Methods
- Deriving the Mean Shift
- Mean shift properties

Applications

- Clustering
- Object Tracking

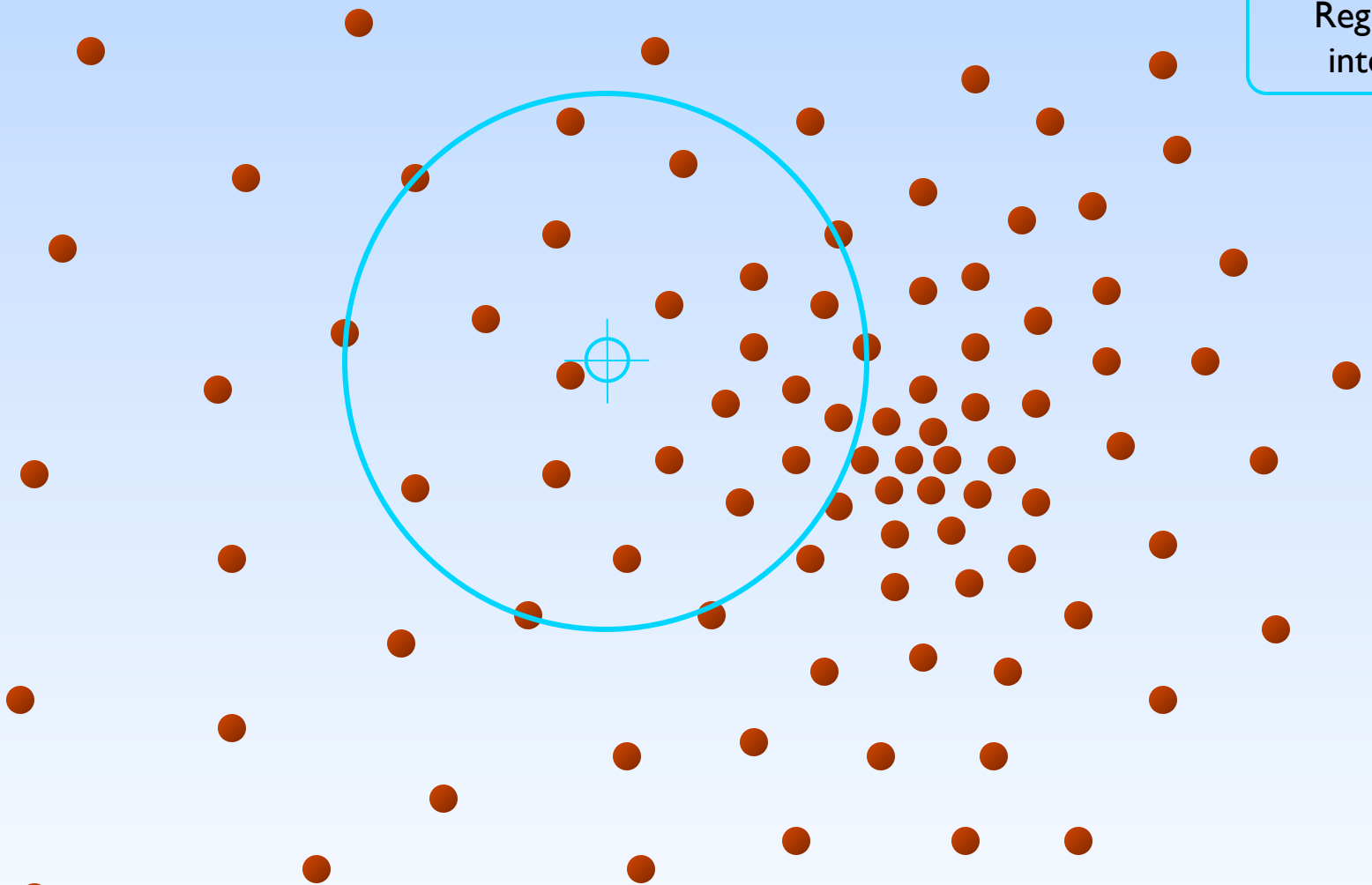
Intuitive Description



Objective : Find the densest region

Distribution of identical billiard balls

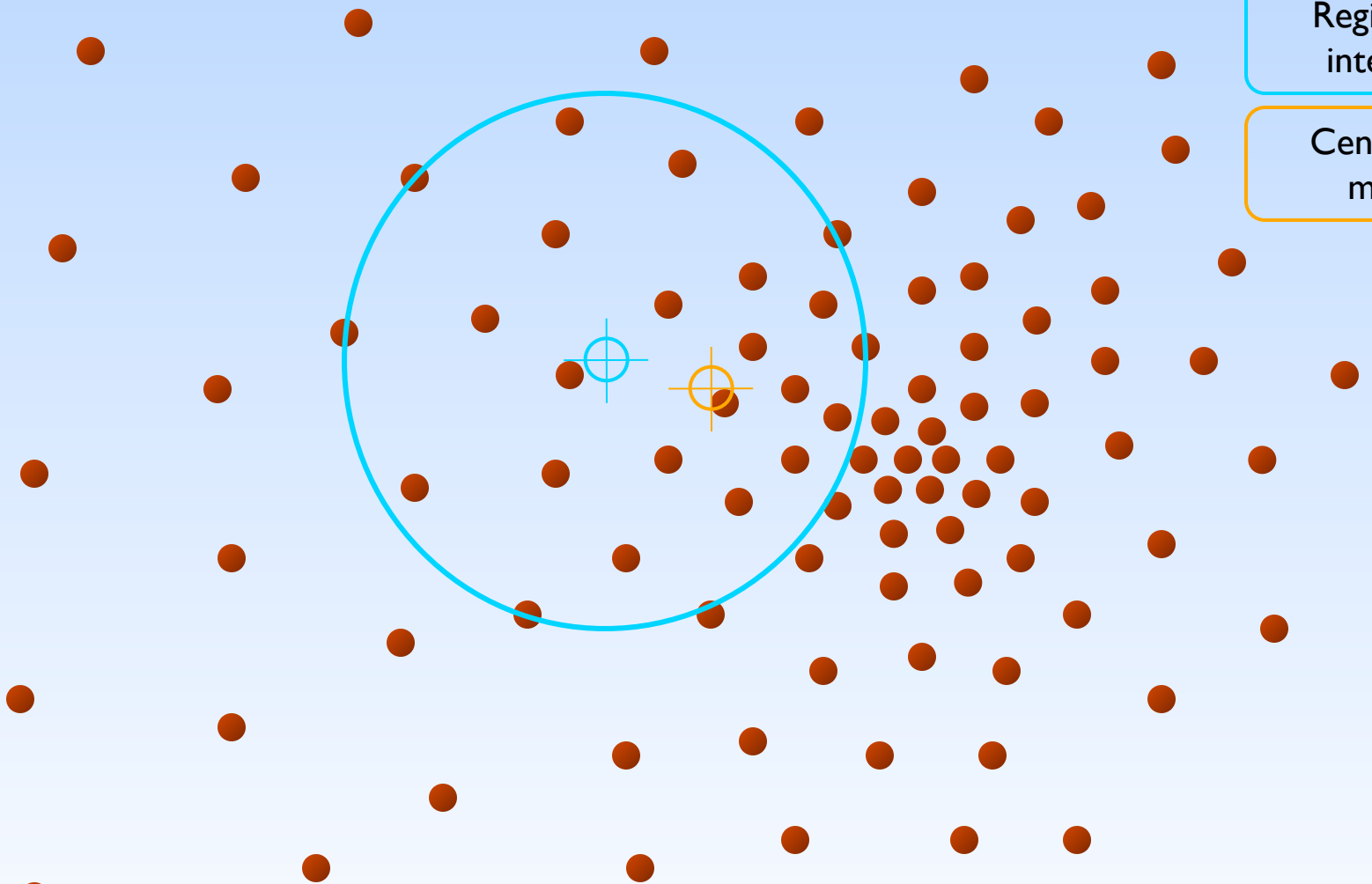
Intuitive Description



Objective : Find the densest region

Distribution of identical billiard balls

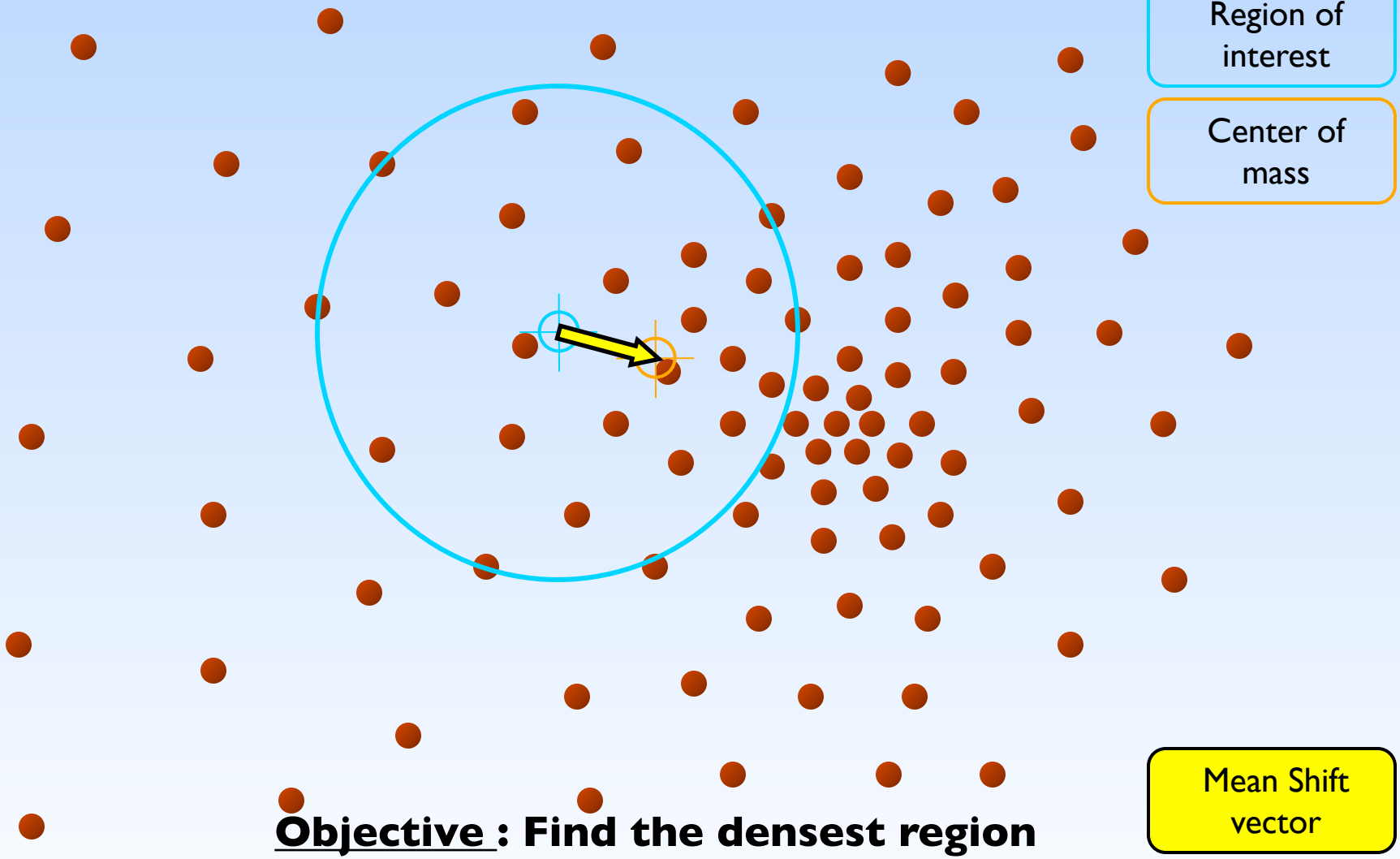
Intuitive Description



Objective : Find the densest region

Distribution of identical billiard balls

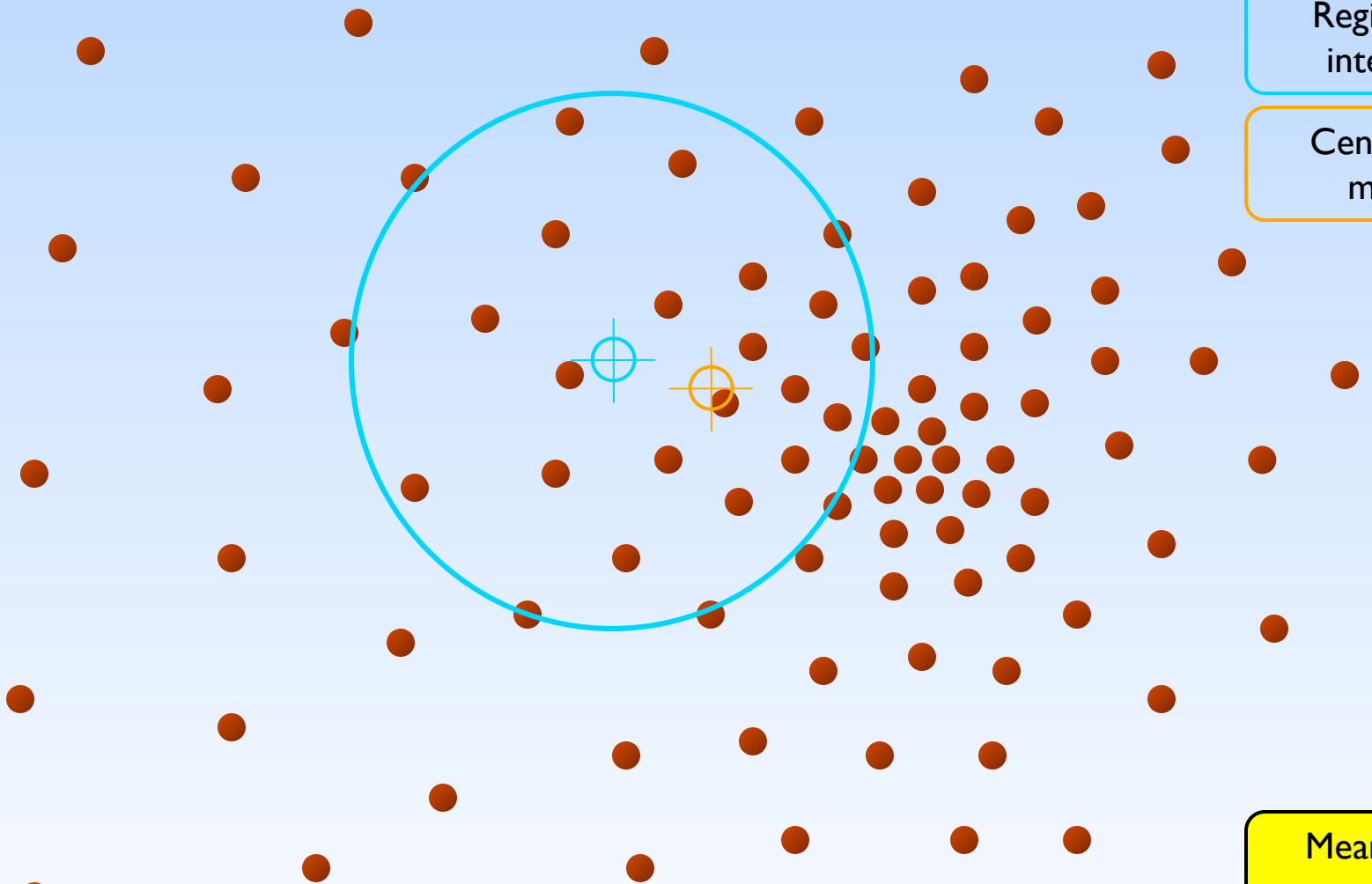
Intuitive Description



Objective : Find the densest region

Distribution of identical billiard balls

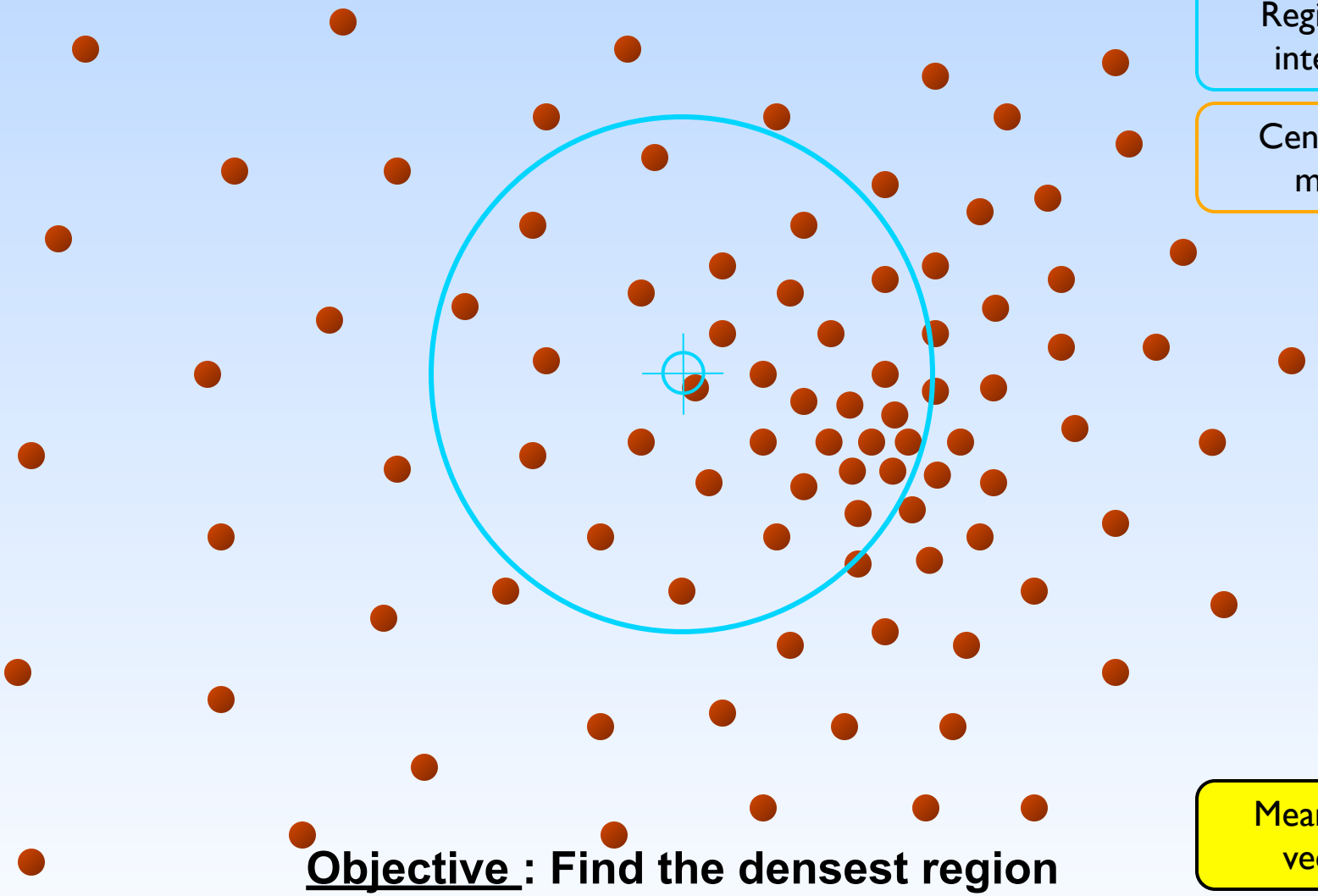
Intuitive Description



Objective : Find the densest region

Distribution of identical billiard balls

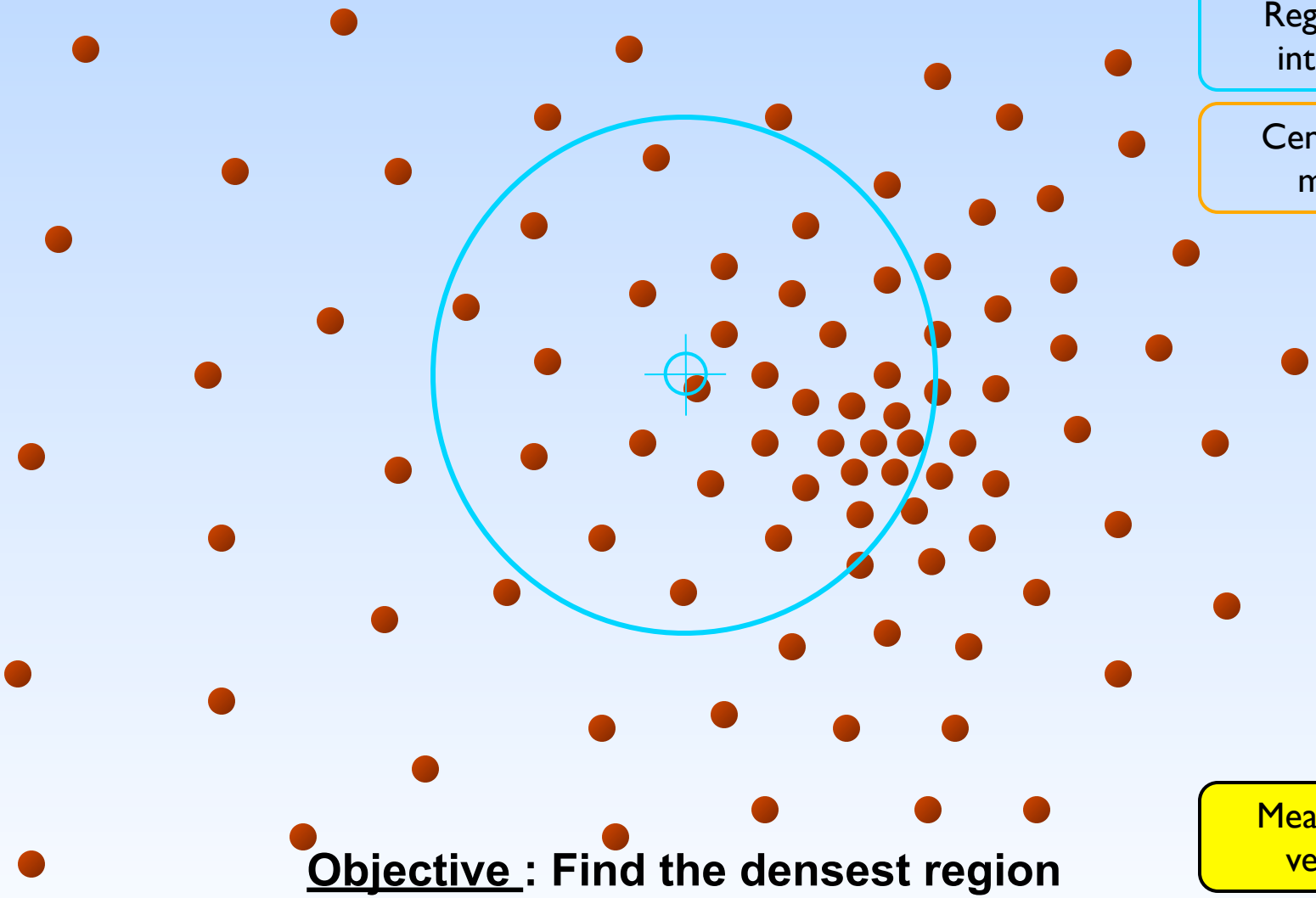
Intuitive Description



Objective : Find the densest region

Distribution of identical billiard balls

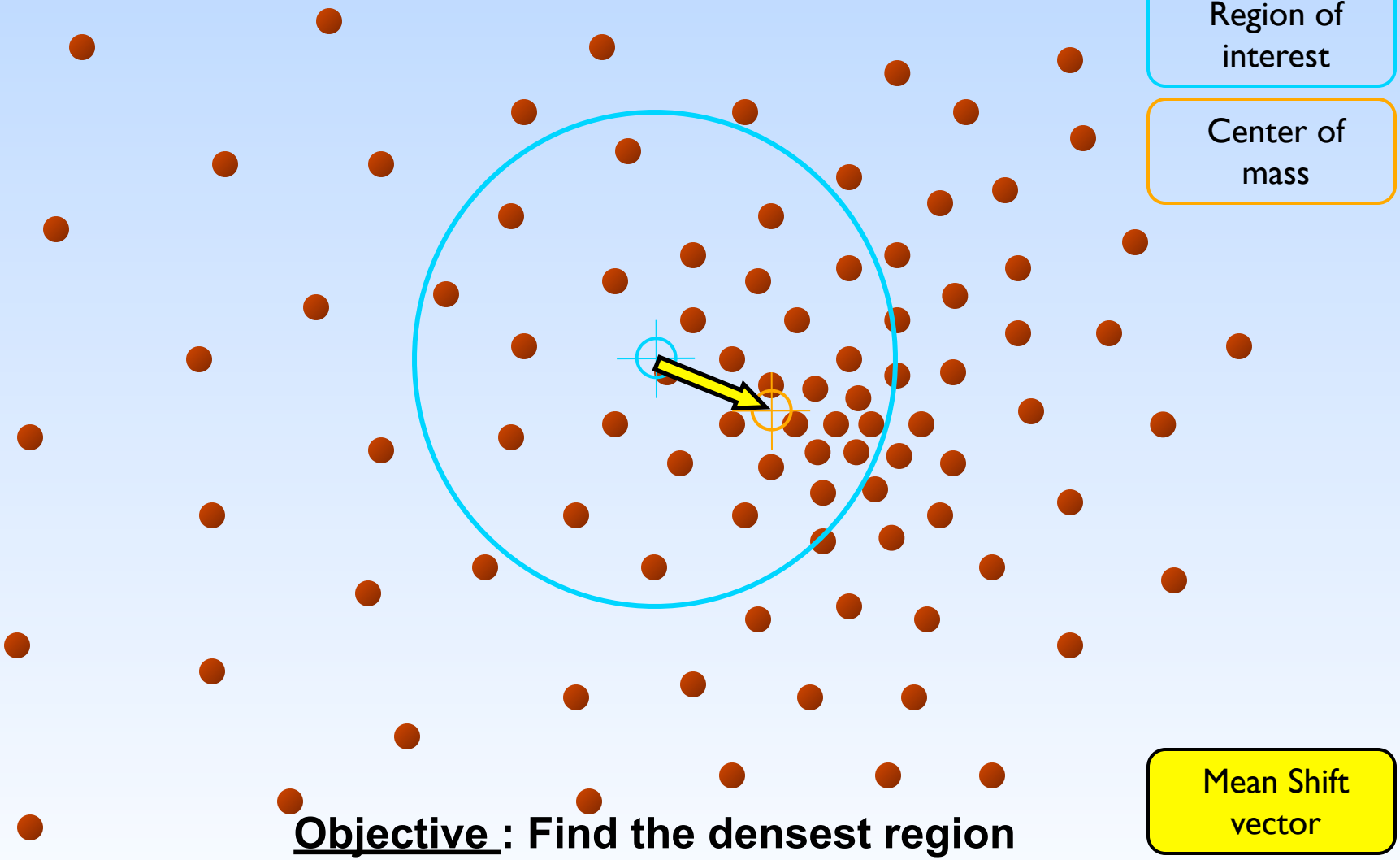
Intuitive Description



Objective : Find the densest region

Distribution of identical billiard balls

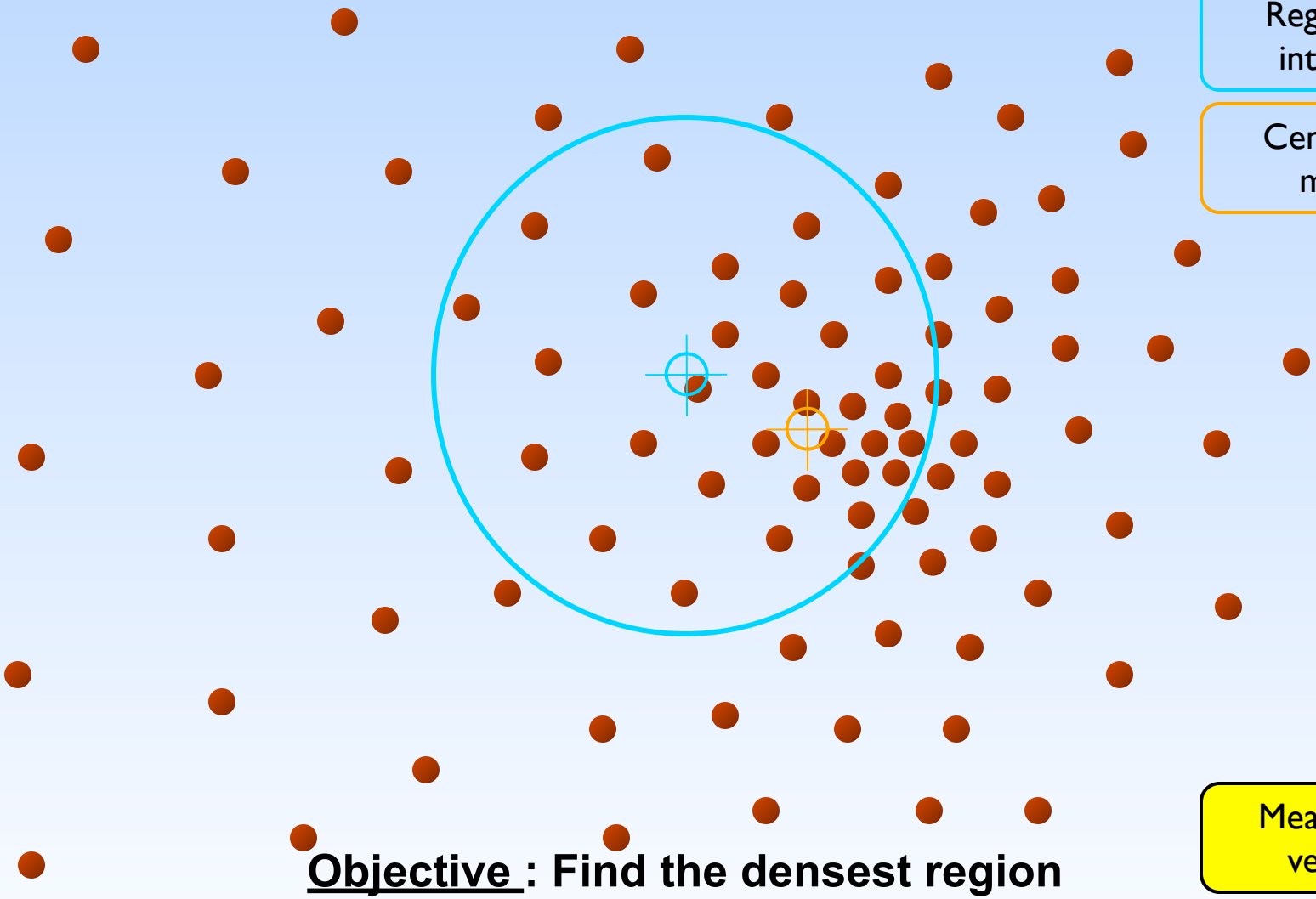
Intuitive Description



Objective : Find the densest region

Distribution of identical billiard balls

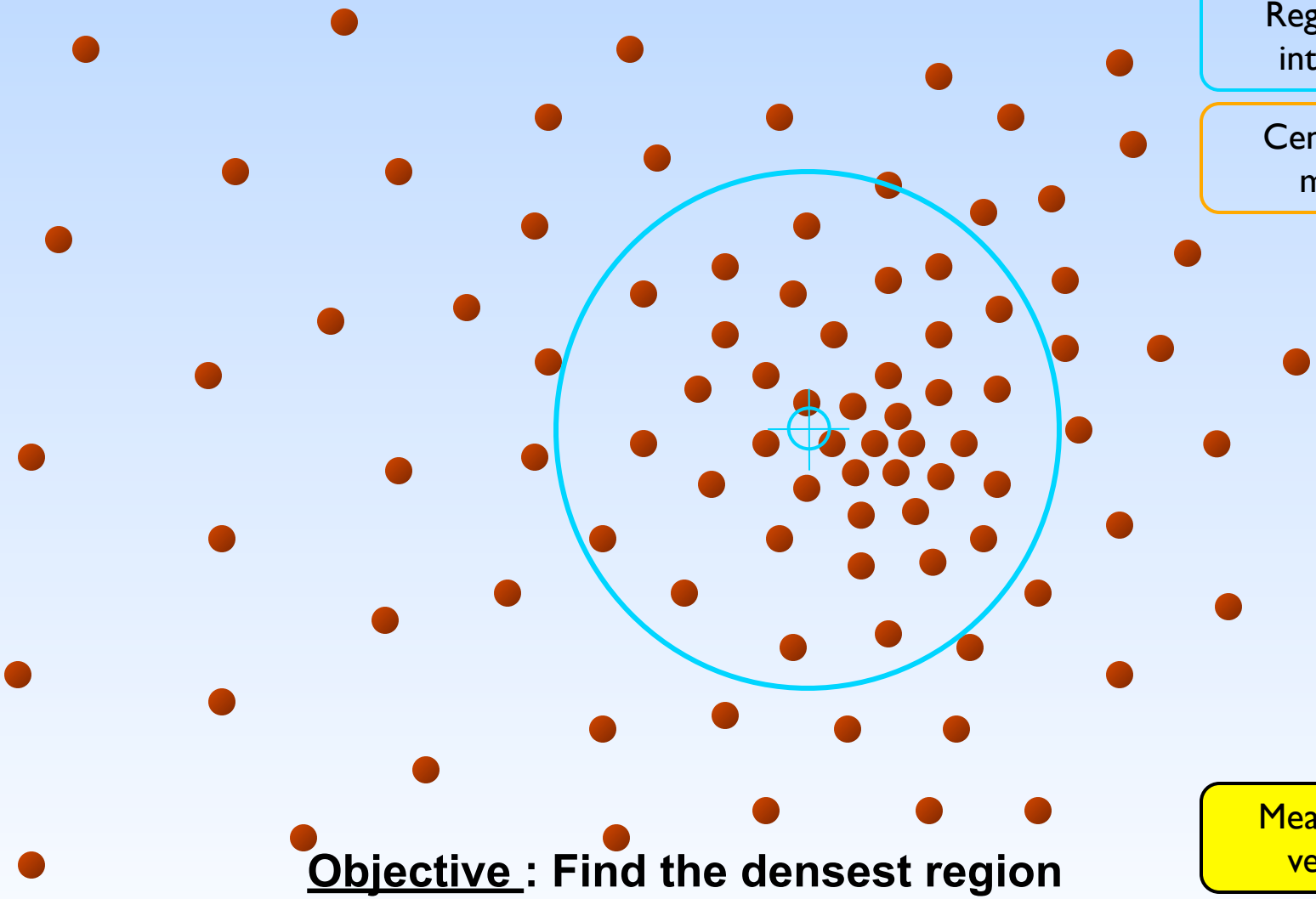
Intuitive Description



Objective : Find the densest region

Distribution of identical billiard balls

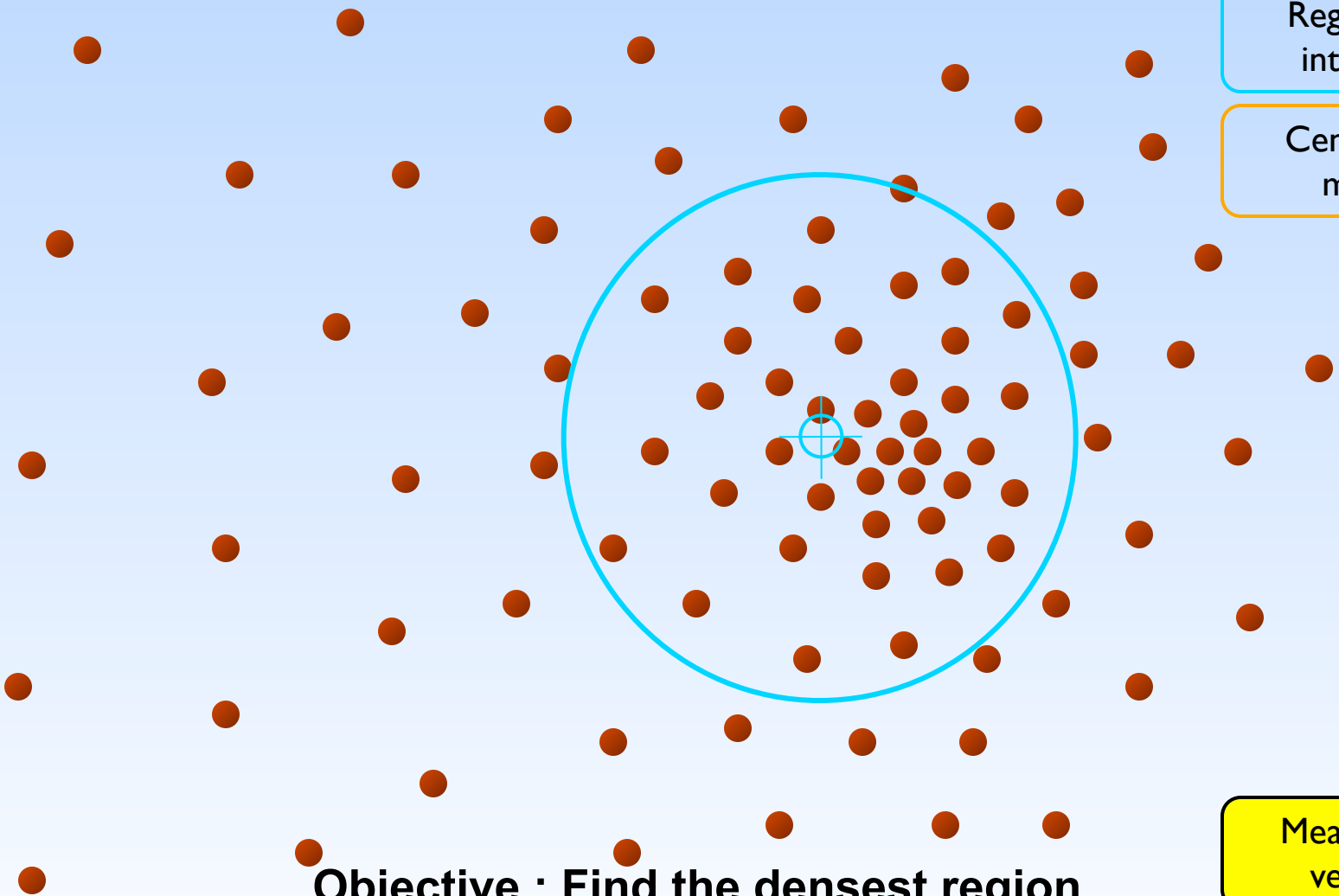
Intuitive Description



Objective : Find the densest region

Distribution of identical billiard balls

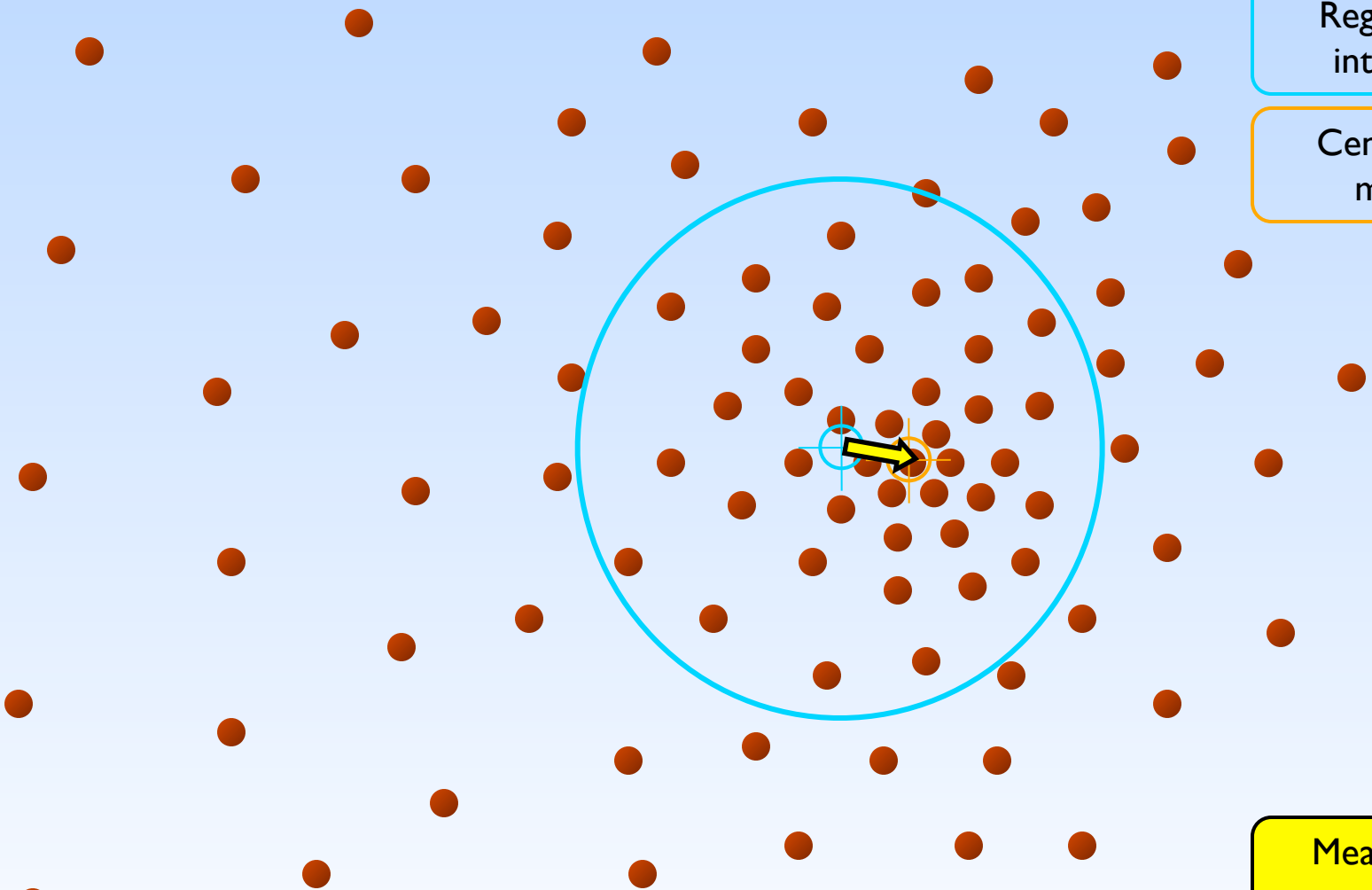
Intuitive Description



Objective : Find the densest region

Distribution of identical billiard balls

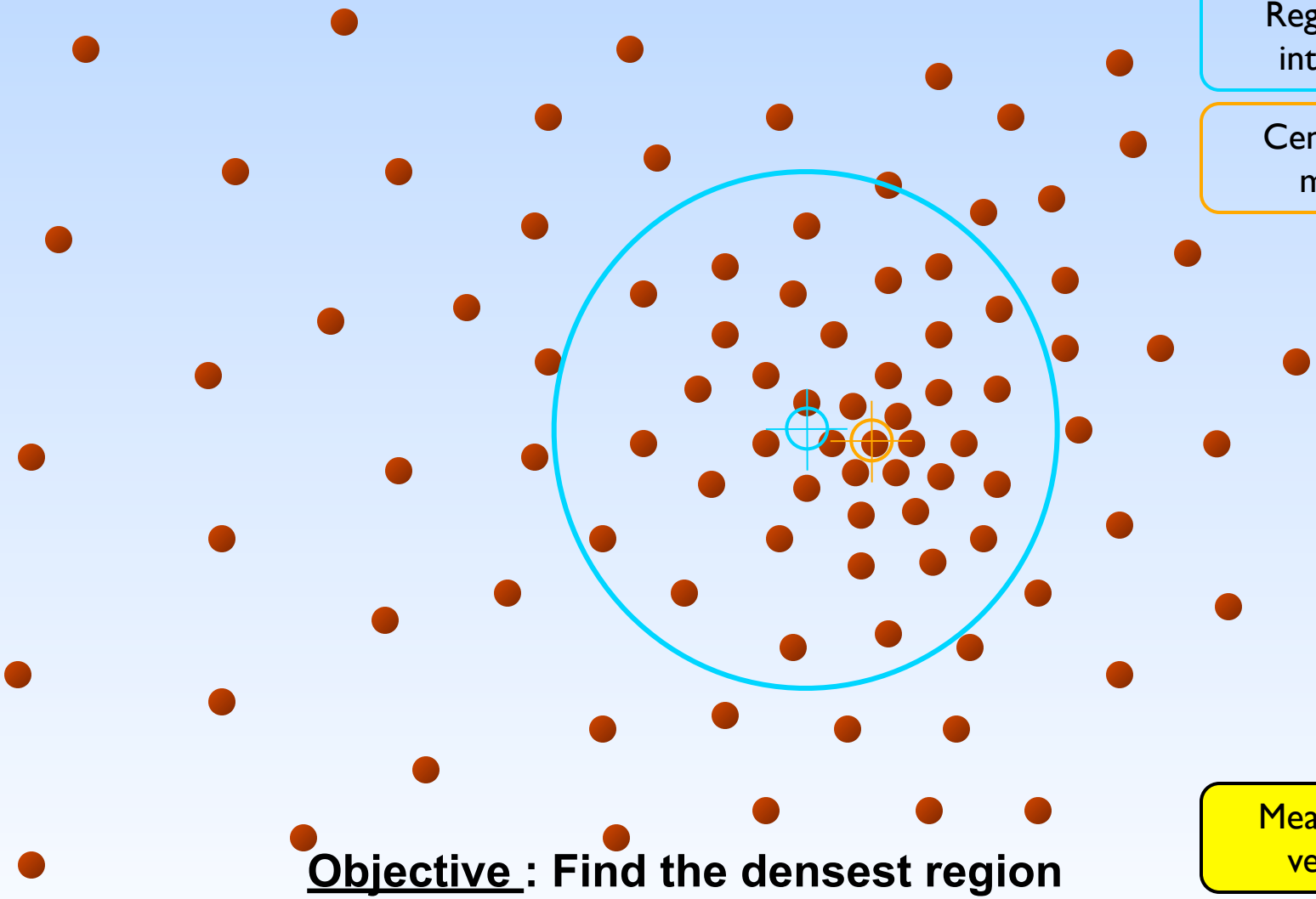
Intuitive Description



Objective : Find the densest region

Distribution of identical billiard balls

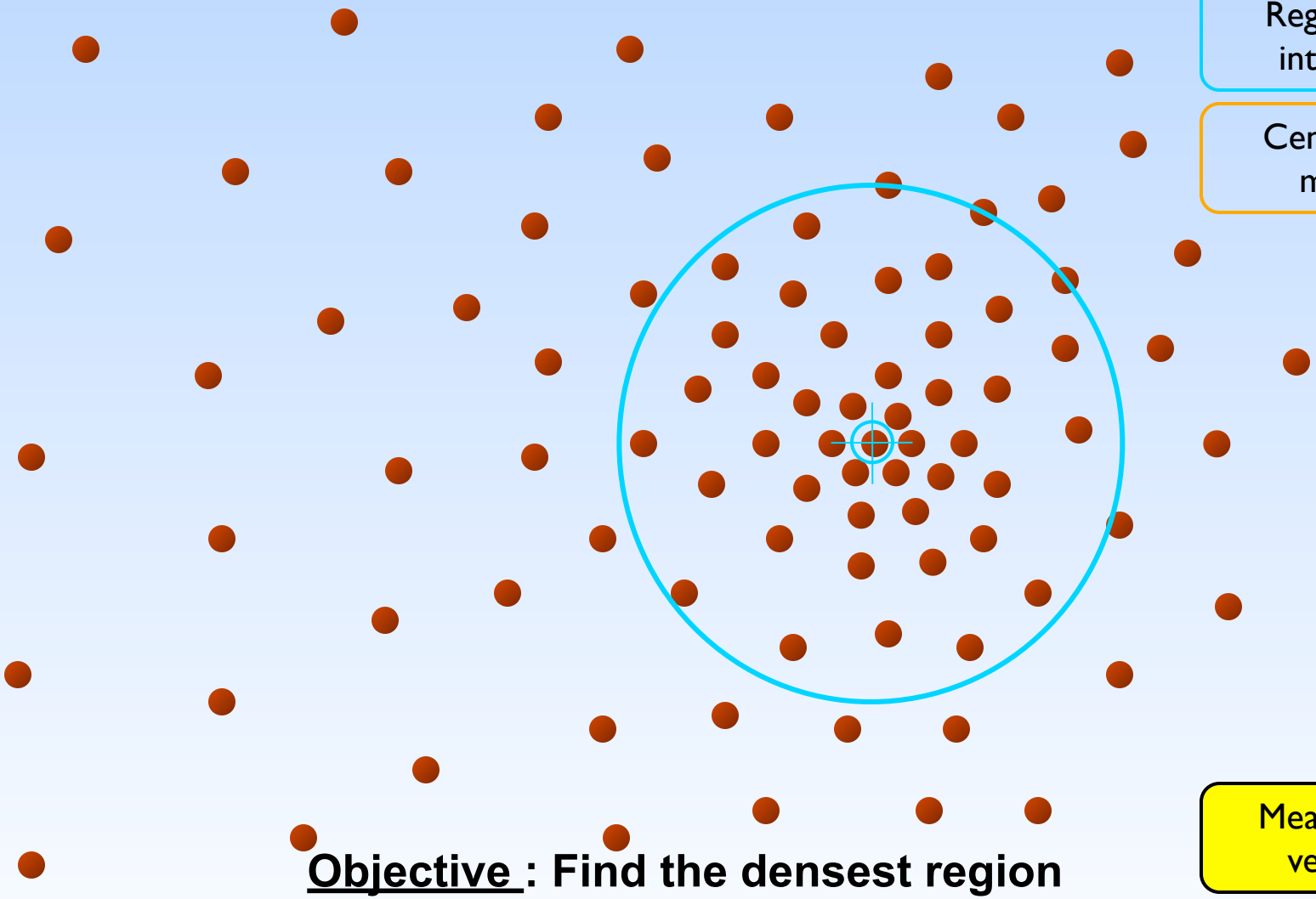
Intuitive Description



Objective : Find the densest region

Distribution of identical billiard balls

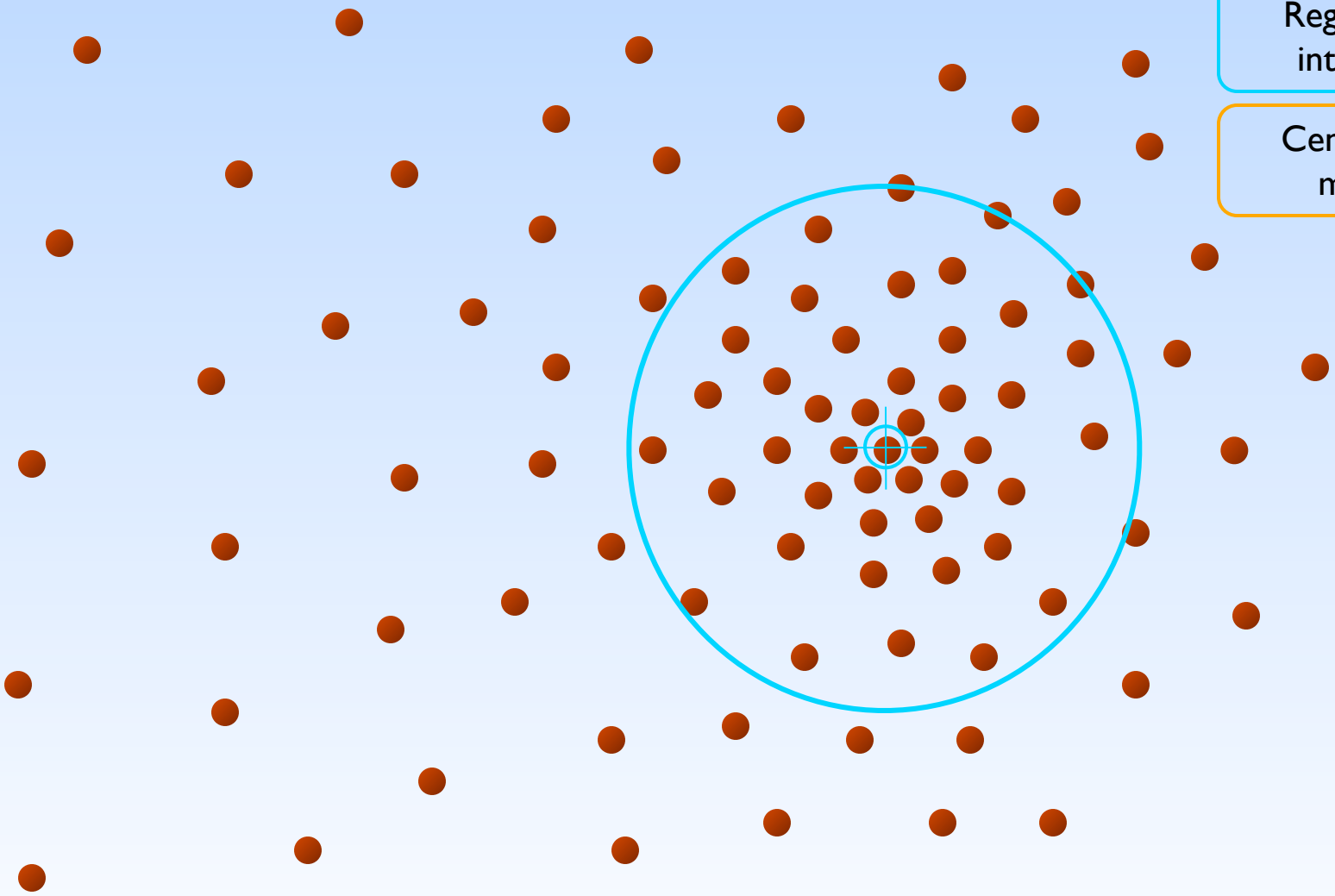
Intuitive Description



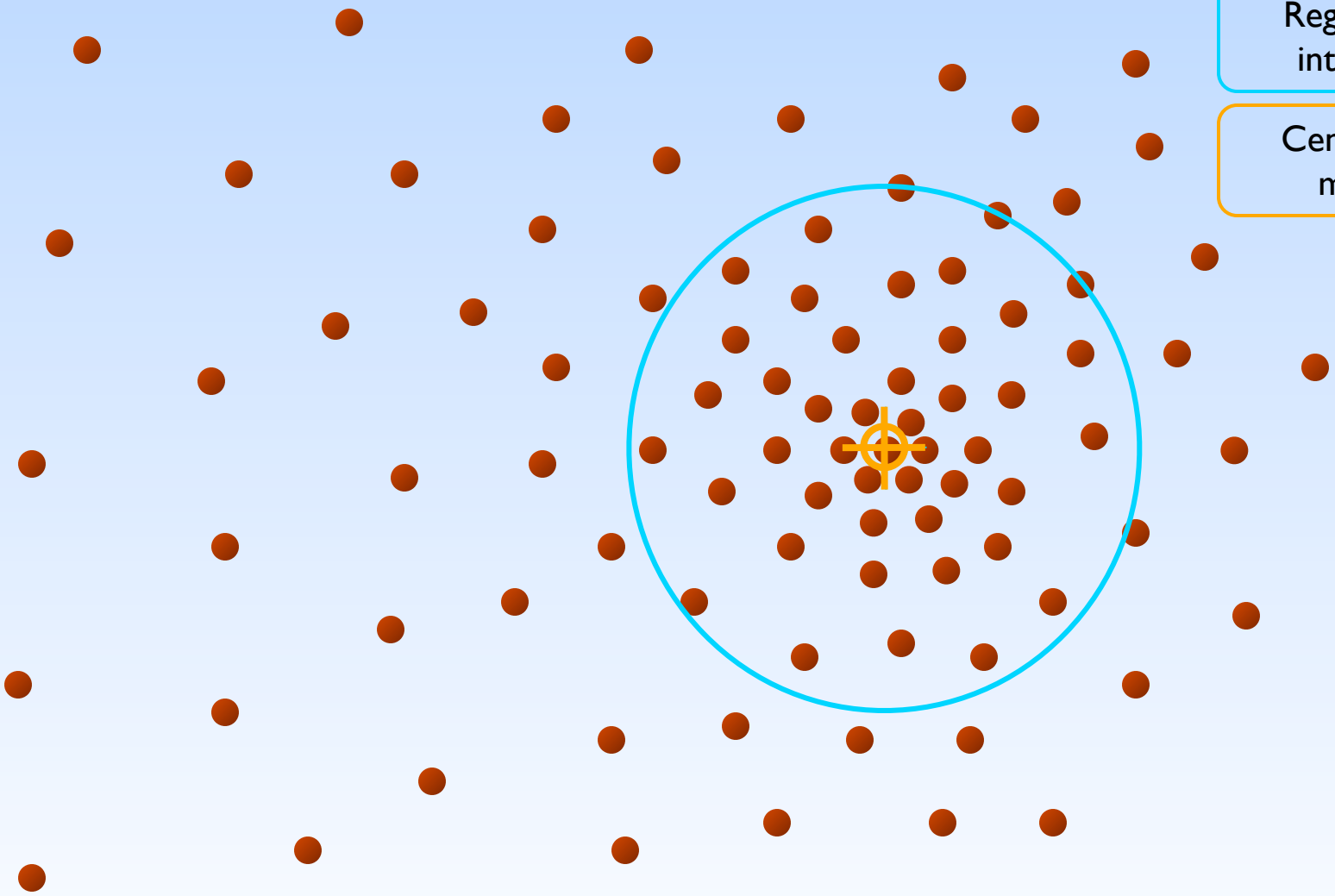
Objective : Find the densest region

Distribution of identical billiard balls

Intuitive Description



Intuitive Description



What is Mean Shift ?

A tool for:
Finding modes in a set of data samples representing an underlying probability density function (PDF) in R^N

What is Mean Shift ?

A tool for:
Finding modes in a set of data samples representing an underlying probability density function (PDF) in R^N

PDF in feature space

What is Mean Shift ?

A tool for:
Finding modes in a set of data samples representing an underlying probability density function (PDF) in R^N

PDF in feature space

- **Color space**

What is Mean Shift ?

A tool for:
Finding modes in a set of data samples representing an underlying probability density function (PDF) in R^N

PDF in feature space

- **Color space**
- **Scale space**

What is Mean Shift ?

A tool for:
Finding modes in a set of data samples representing an underlying probability density function (PDF) in R^N

PDF in feature space

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

What is Mean Shift ?

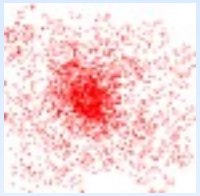
A tool for:
Finding modes in a set of data samples representing an underlying probability density function (PDF) in R^N

PDF in feature space

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

What is Mean Shift ?

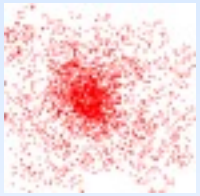
A tool for:
Finding modes in a set of data samples representing an underlying probability density function (PDF) in R^N



Data

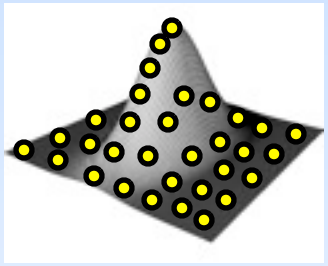
What is Mean Shift ?

A tool for:
Finding modes in a set of data samples representing an underlying probability density function (PDF) in R^N



Data

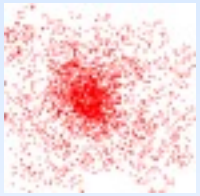
Non-parametric Density Estimation



Discrete PDF Representation

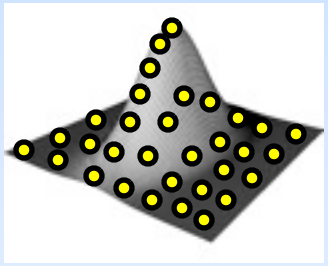
What is Mean Shift ?

A tool for:
Finding modes in a set of data samples representing an underlying probability density function (PDF) in R^N



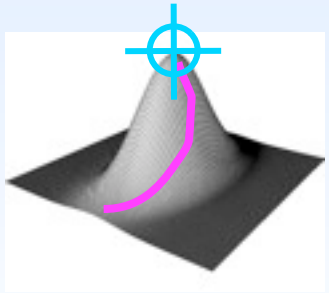
Data

Non-parametric
Density Estimation



Discrete PDF Representation

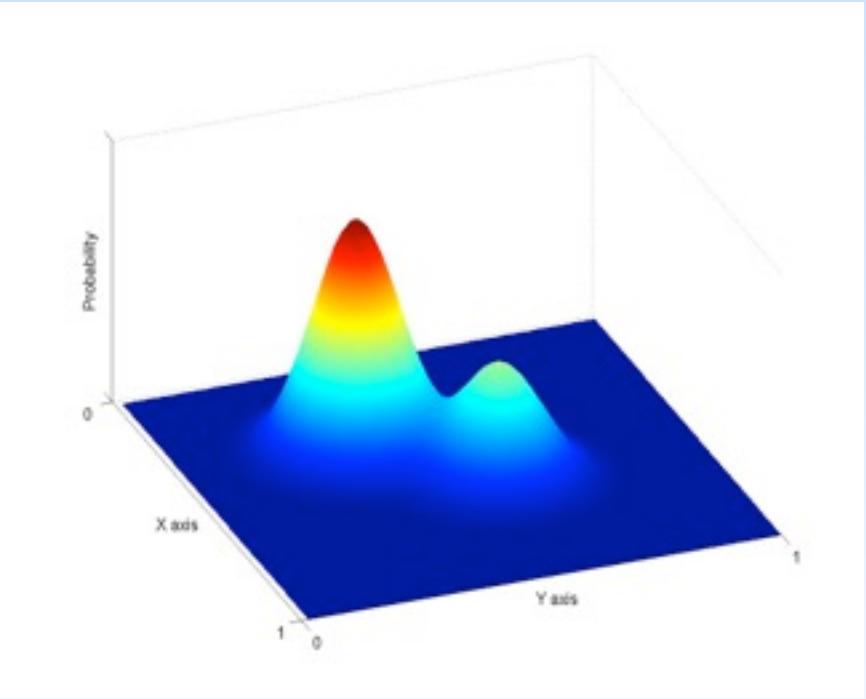
Non-parametric
Density **GRADIENT** Estimation
(Mean Shift)

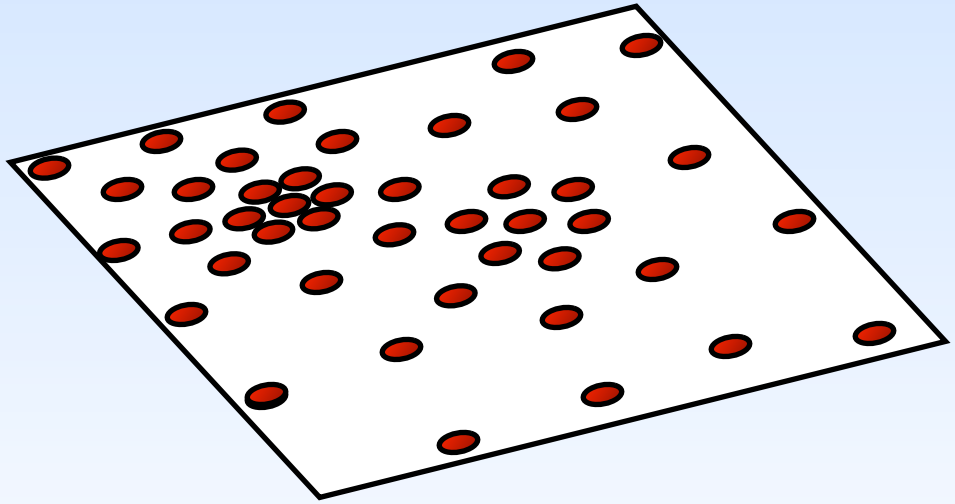


PDF Analysis

Non-Parametric Density Estimation

Assumption : The data points are sampled from an underlying PDF



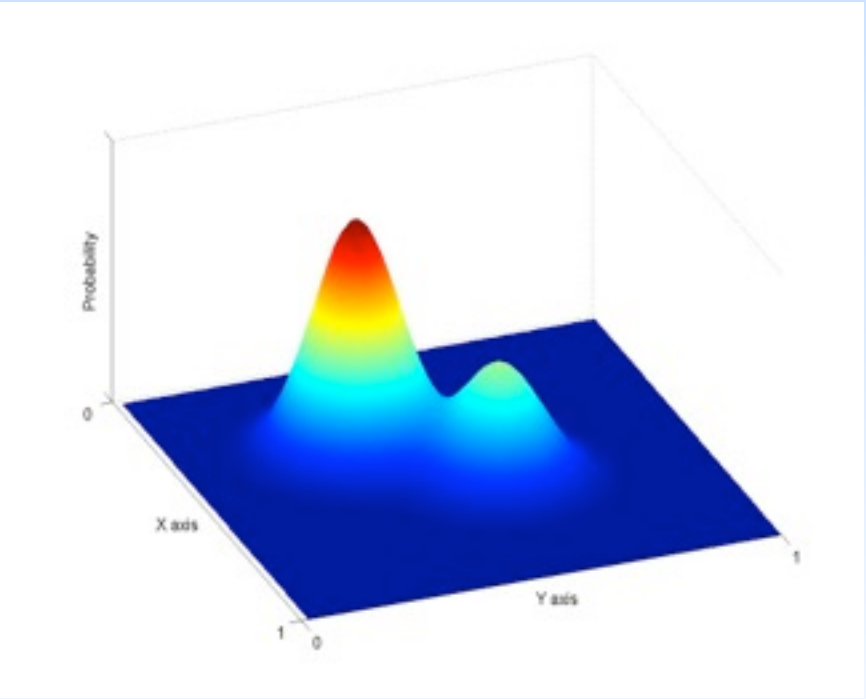


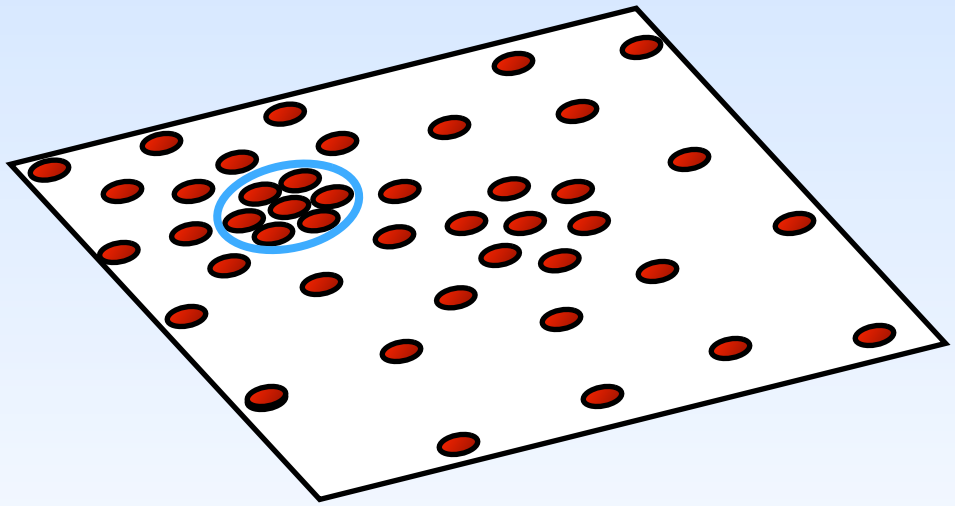
Assumed Underlying PDF

Real Data Samples

Non-Parametric Density Estimation

Assumption : The data points are sampled from an underlying PDF



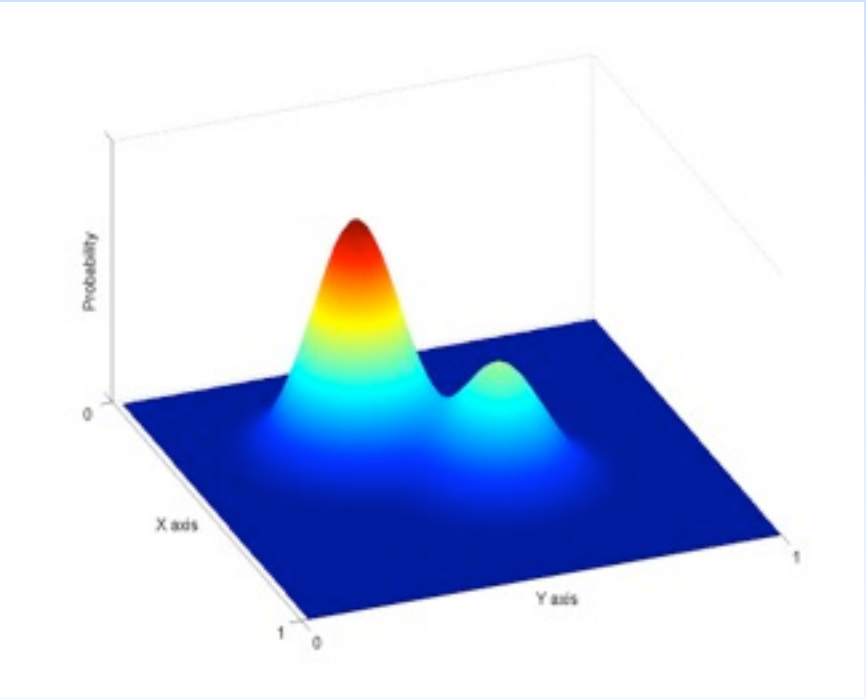


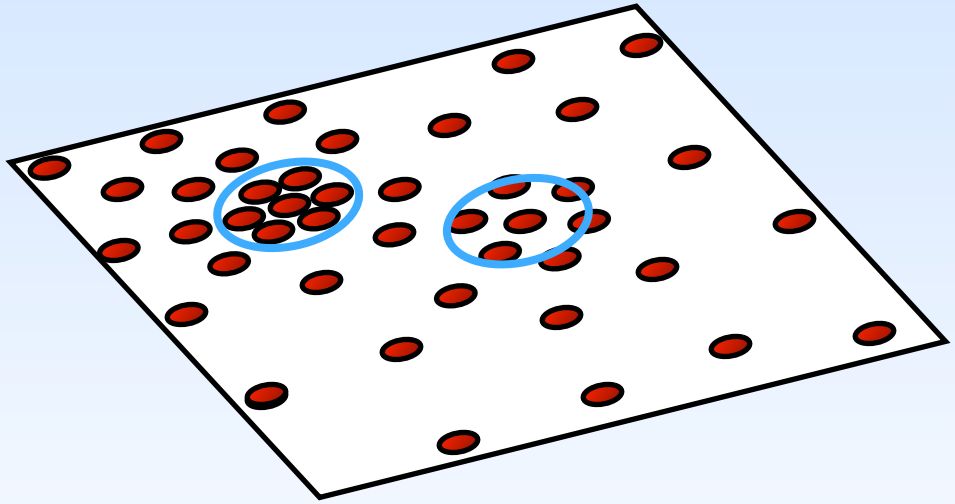
Assumed Underlying PDF

Real Data Samples

Non-Parametric Density Estimation

Assumption : The data points are sampled from an underlying PDF



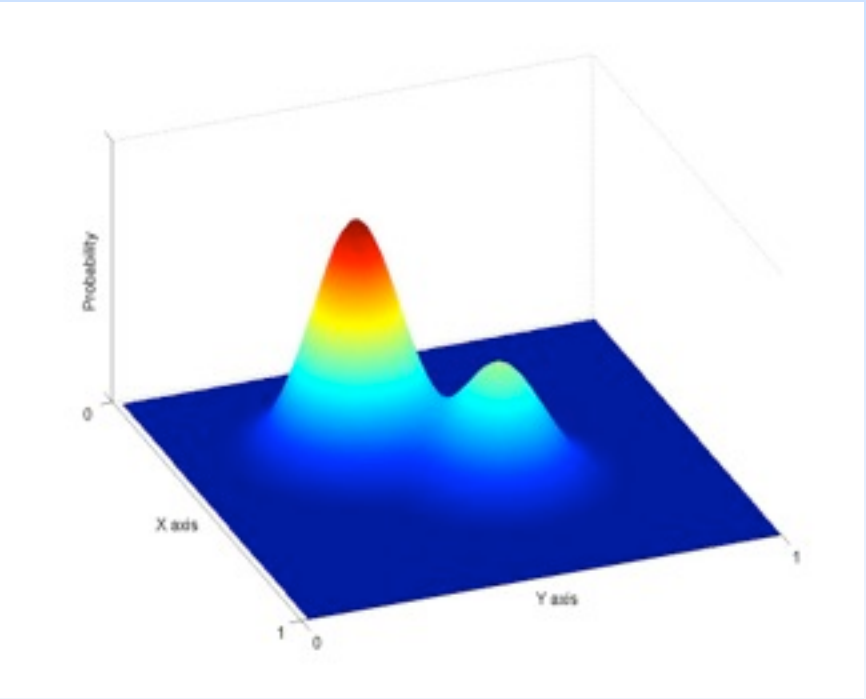


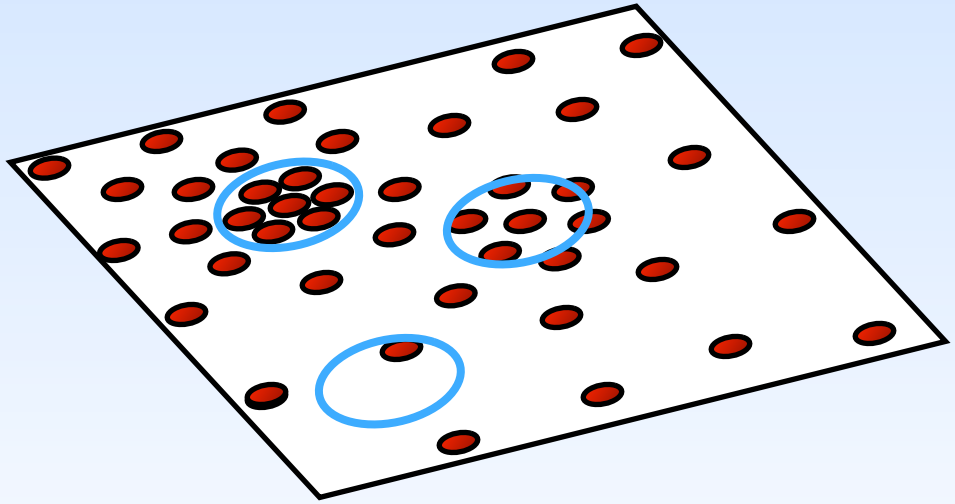
Assumed Underlying PDF

Real Data Samples

Non-Parametric Density Estimation

Assumption : The data points are sampled from an underlying PDF



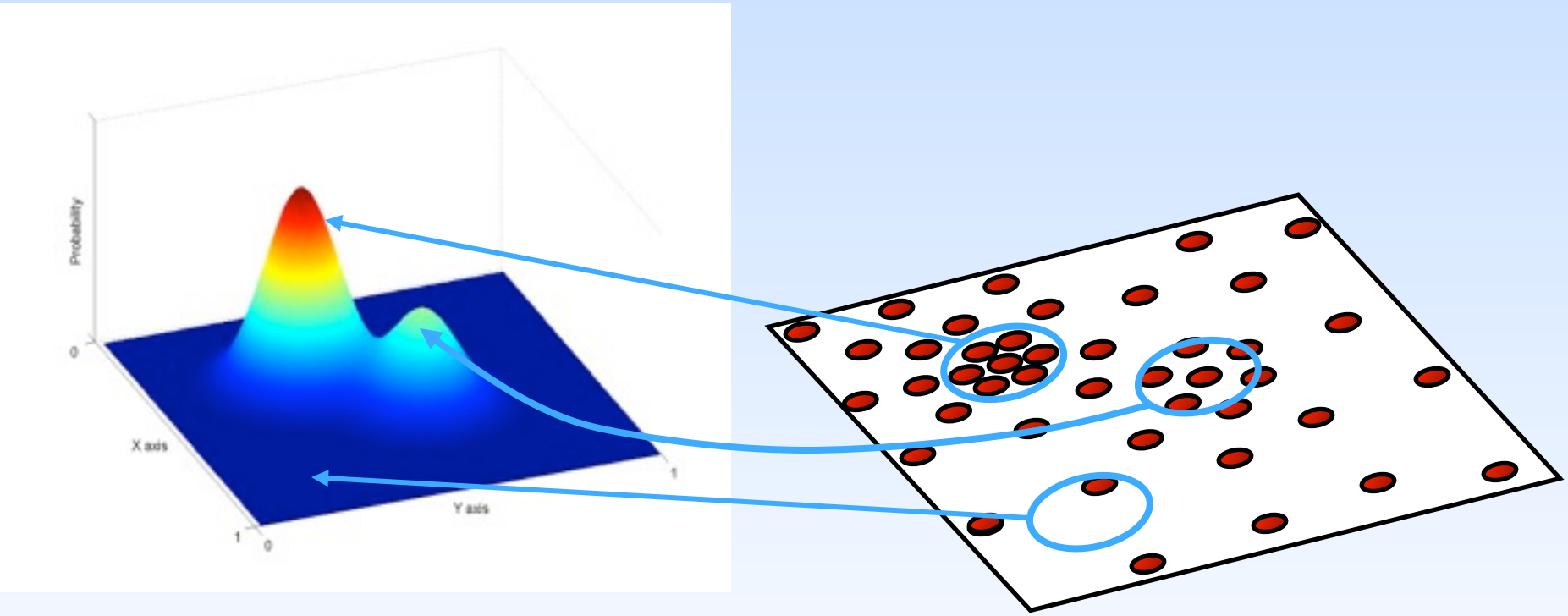


Assumed Underlying PDF

Real Data Samples

Non-Parametric Density Estimation

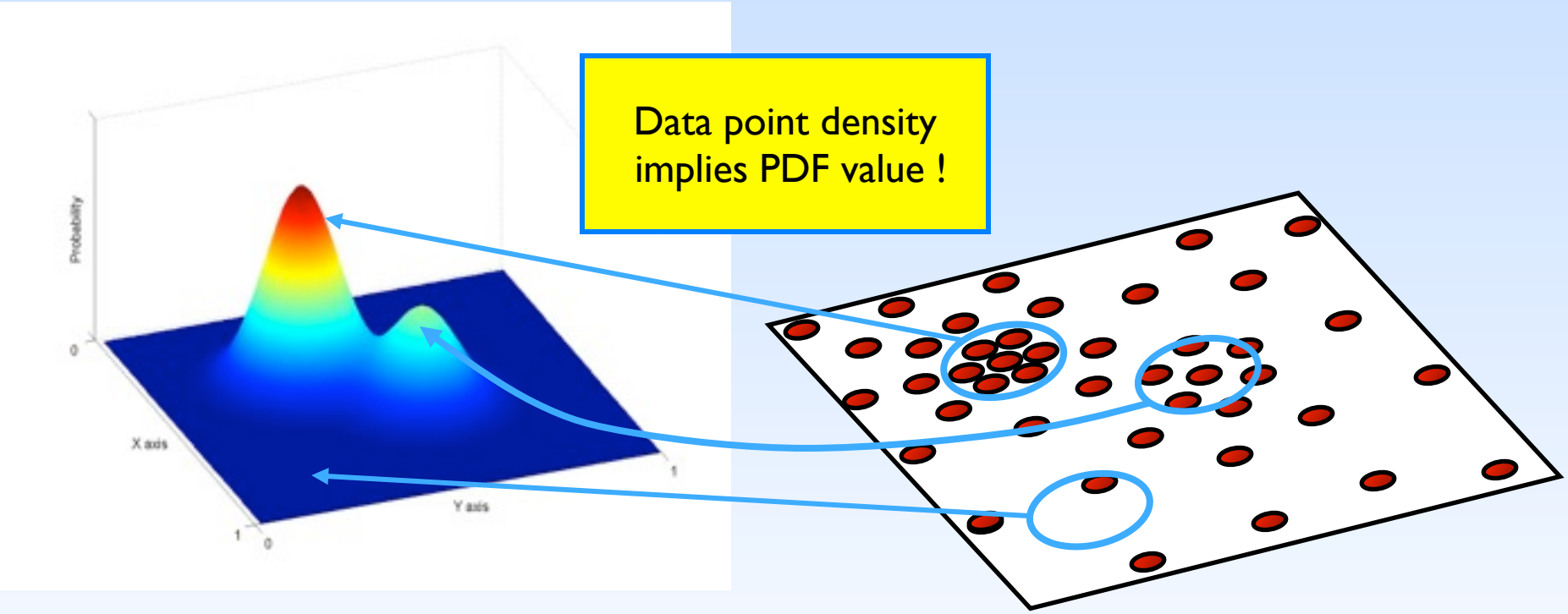
Assumption : The data points are sampled from an underlying PDF



Assumed Underlying PDF **Real Data Samples**

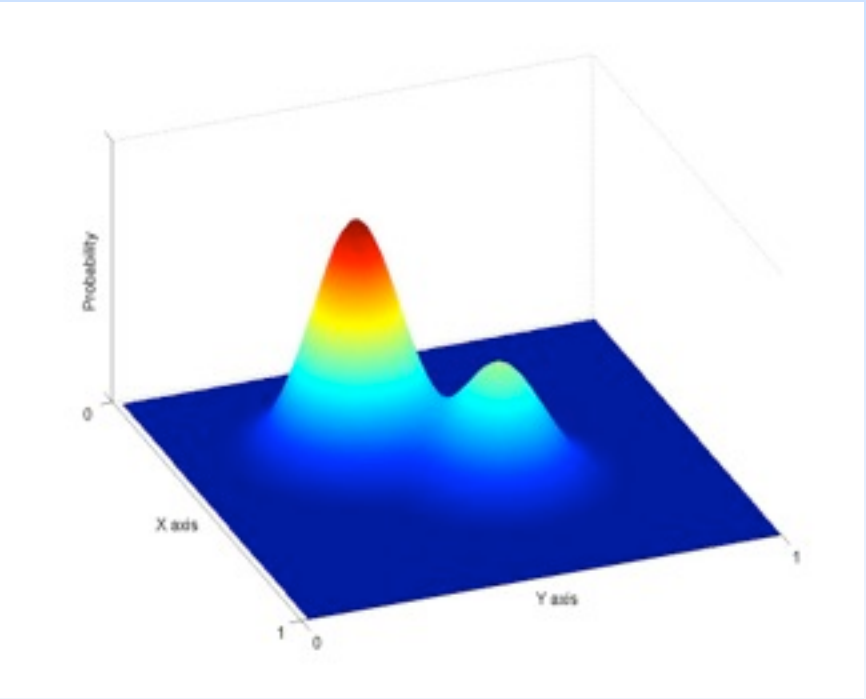
Non-Parametric Density Estimation

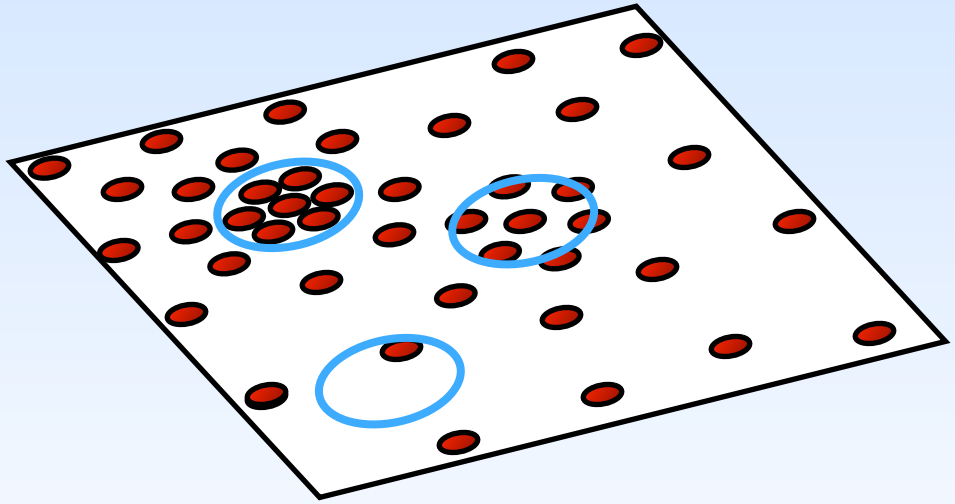
Assumption : The data points are sampled from an underlying PDF



Assumed Underlying PDF **Real Data Samples**

Non-Parametric Density Estimation

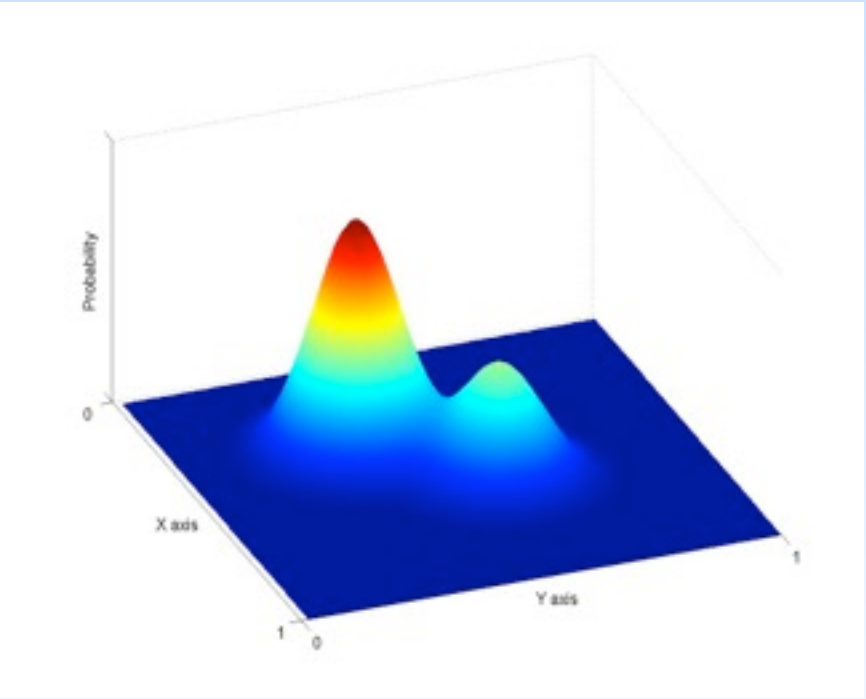


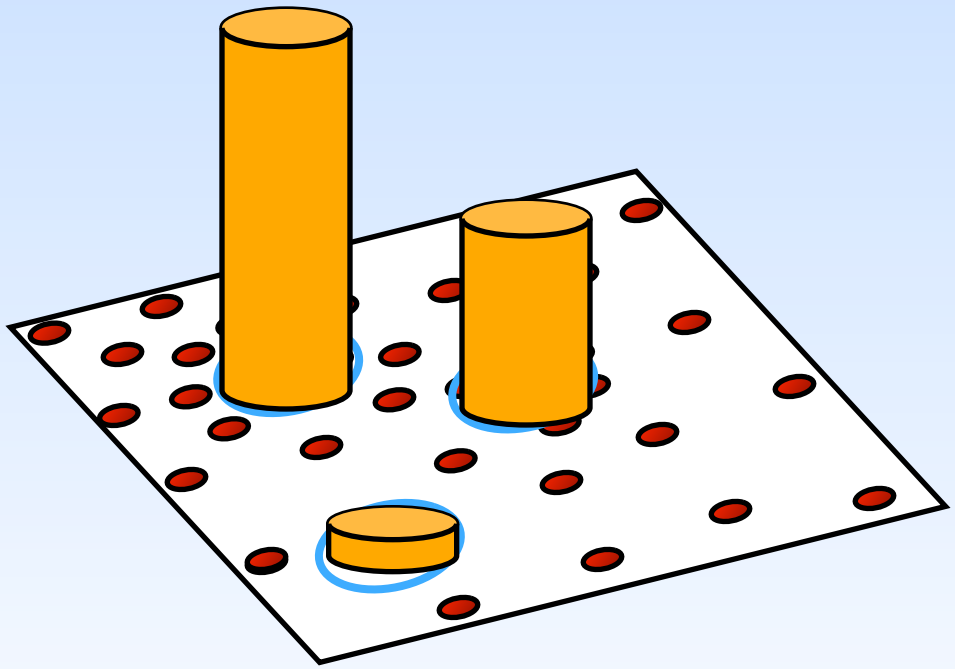


Assumed Underlying PDF

Real Data Samples

Non-Parametric Density Estimation

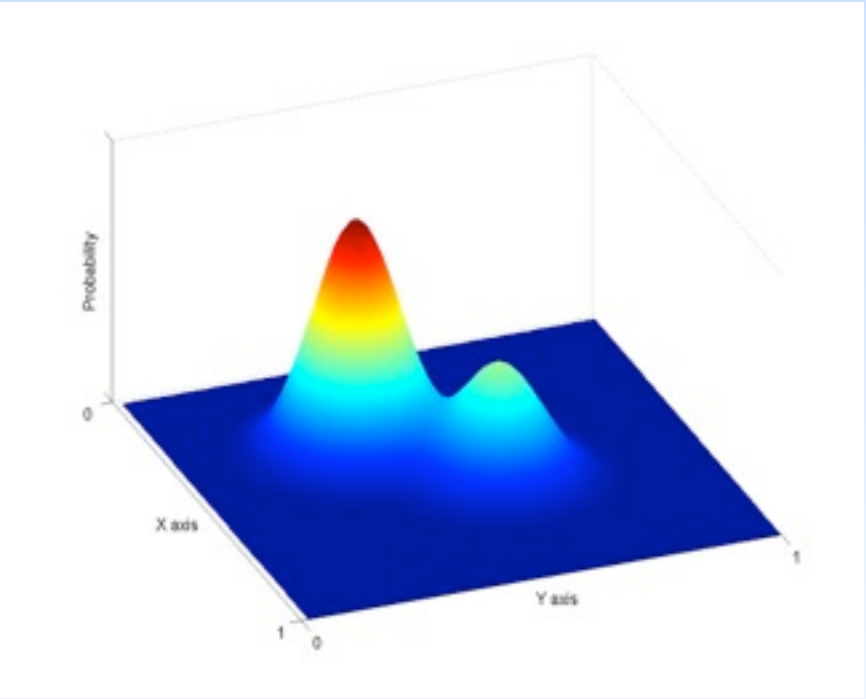


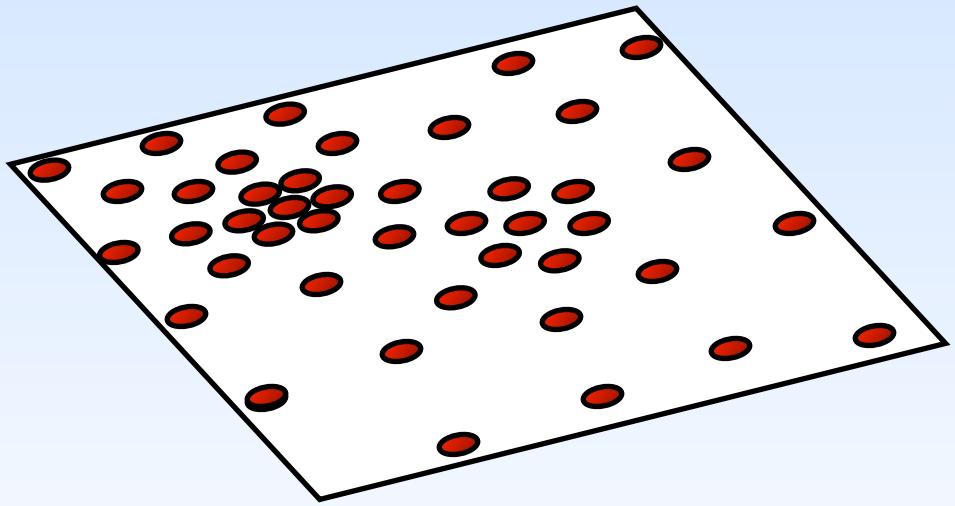


Assumed Underlying PDF

Real Data Samples

Non-Parametric Density Estimation

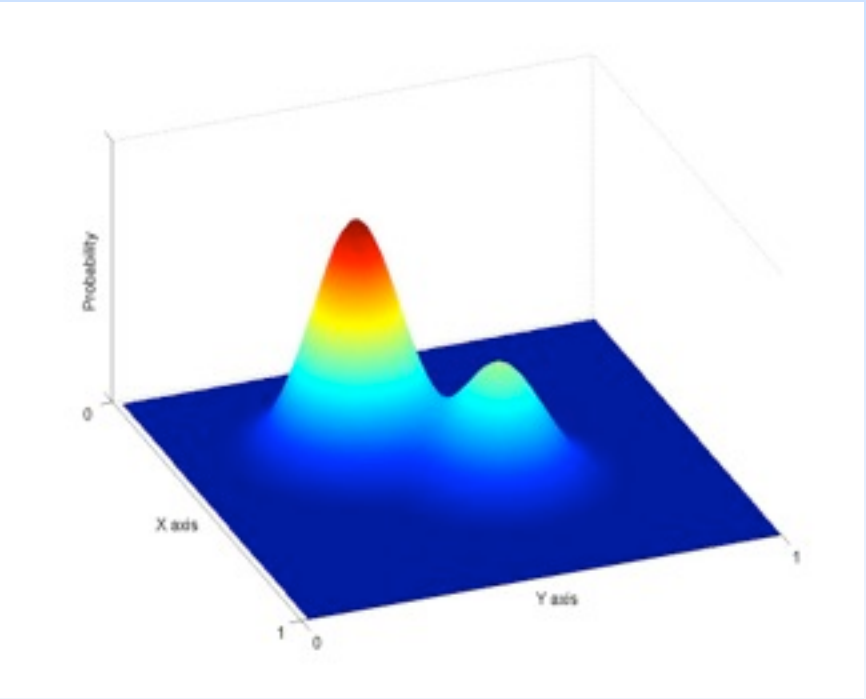


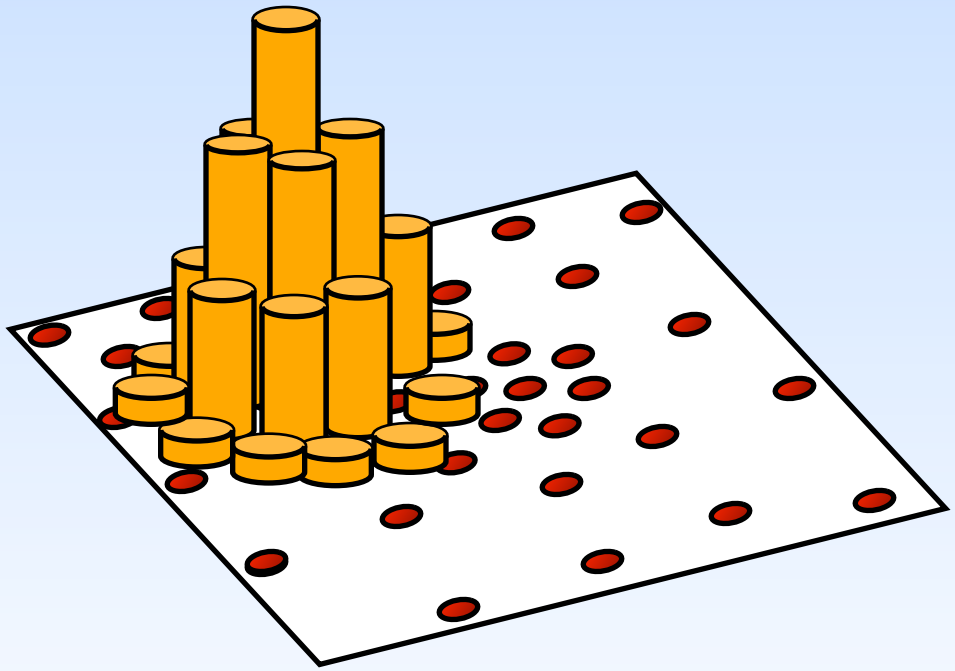


Assumed Underlying PDF

Real Data Samples

Non-Parametric Density Estimation

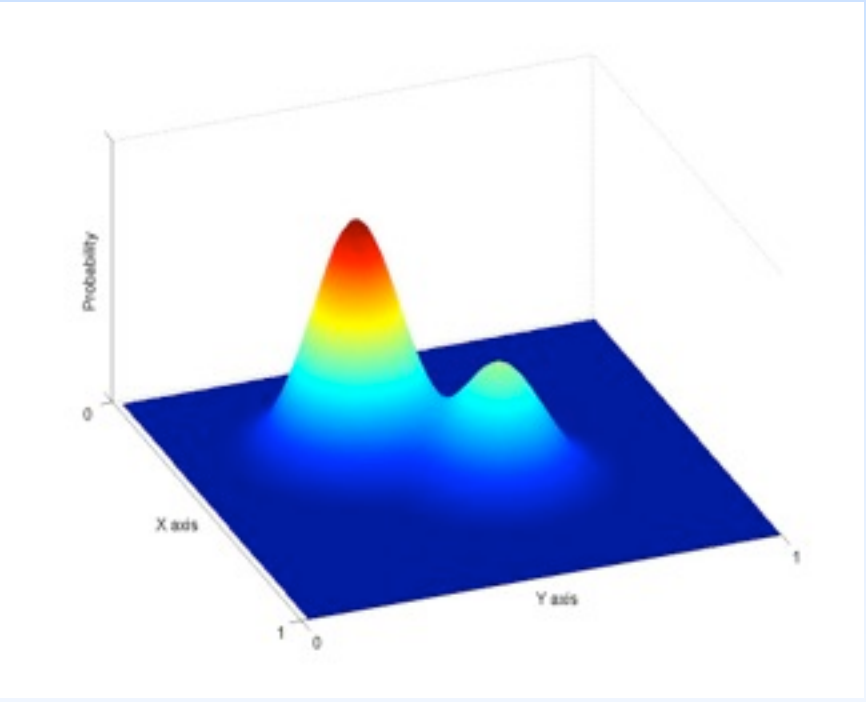


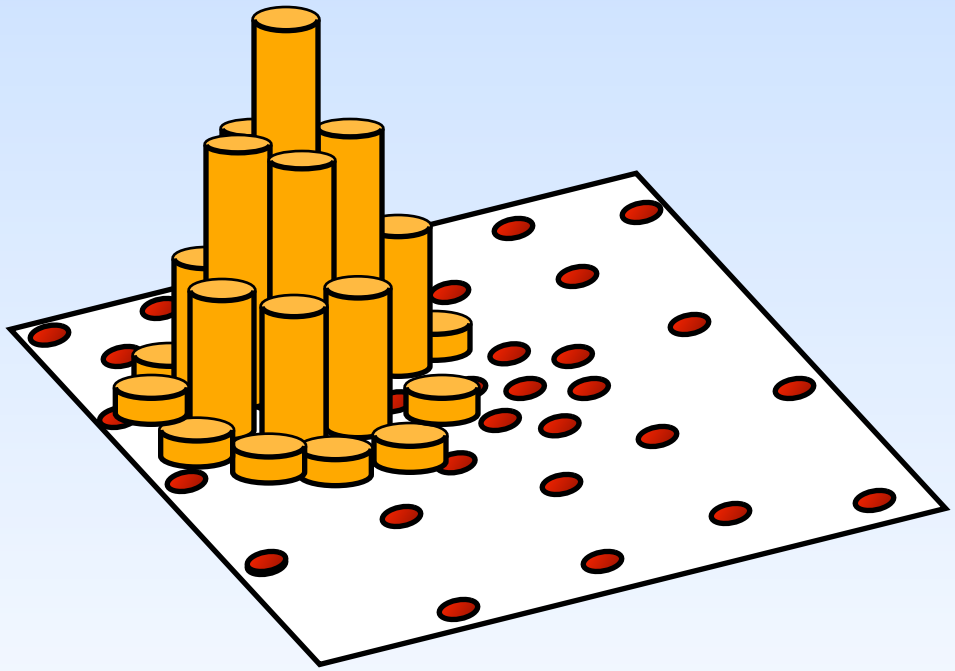


Assumed Underlying PDF

Real Data Samples

Non-Parametric Density Estimation



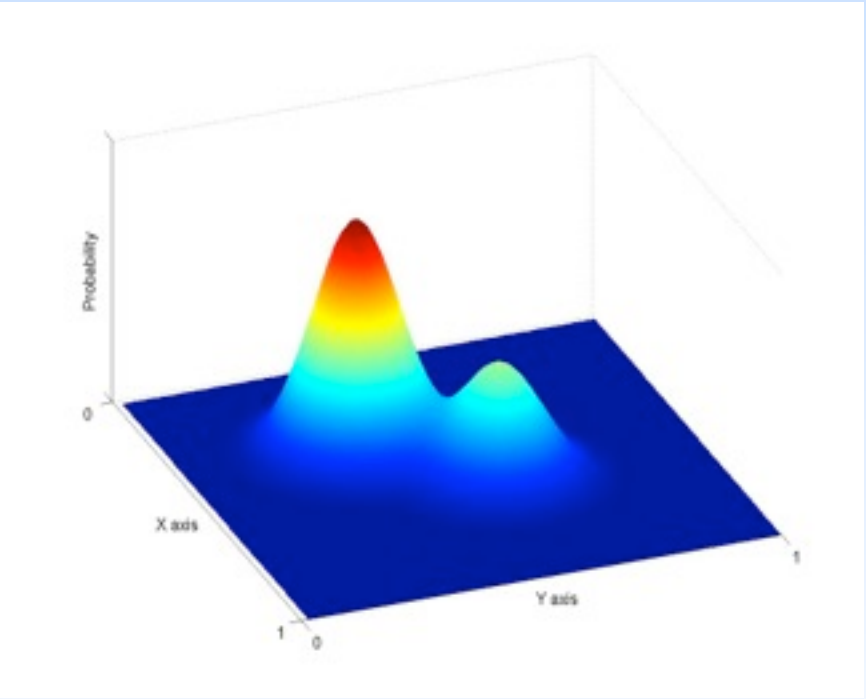


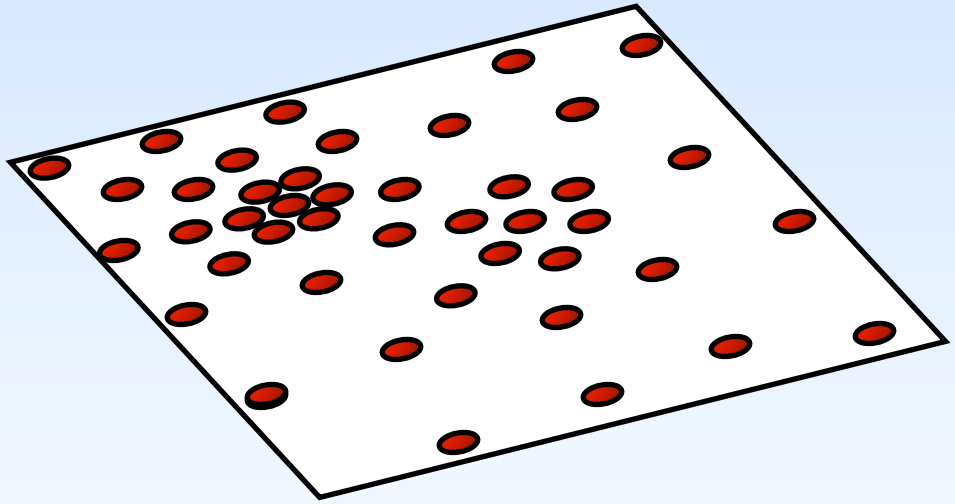
Assumed Underlying PDF

Real Data Samples

Parametric Density

Assumption : The data points are sampled from an underlying PDF





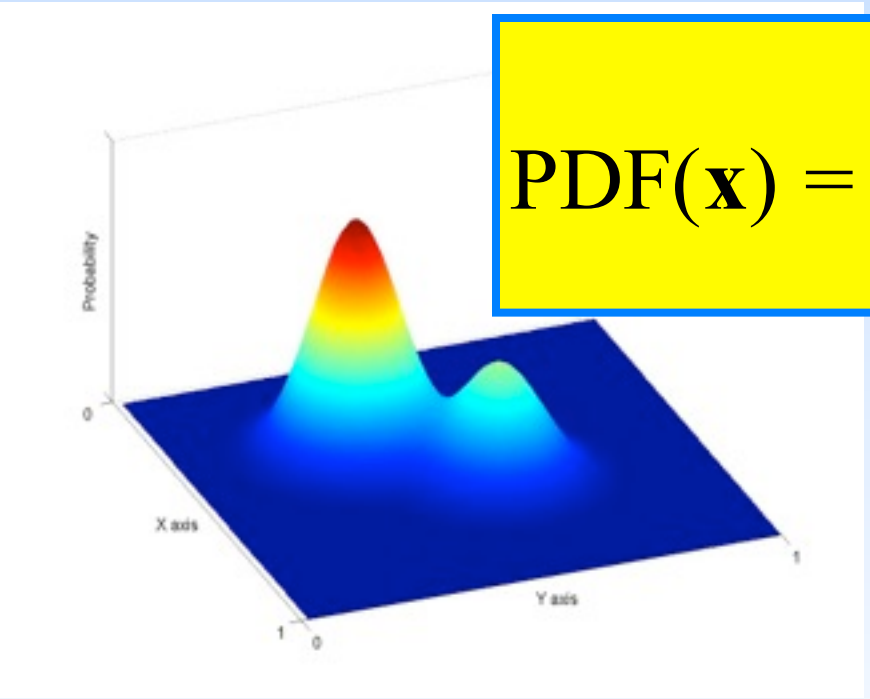
Assumed Underlying PDF

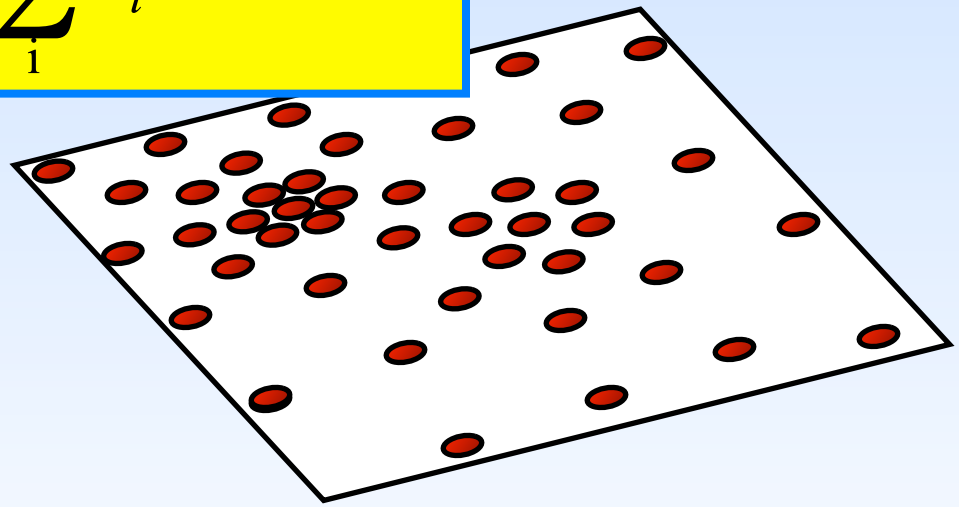
Real Data Samples

Parametric Density

Assumption : The data points are sampled from an underlying PDF

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





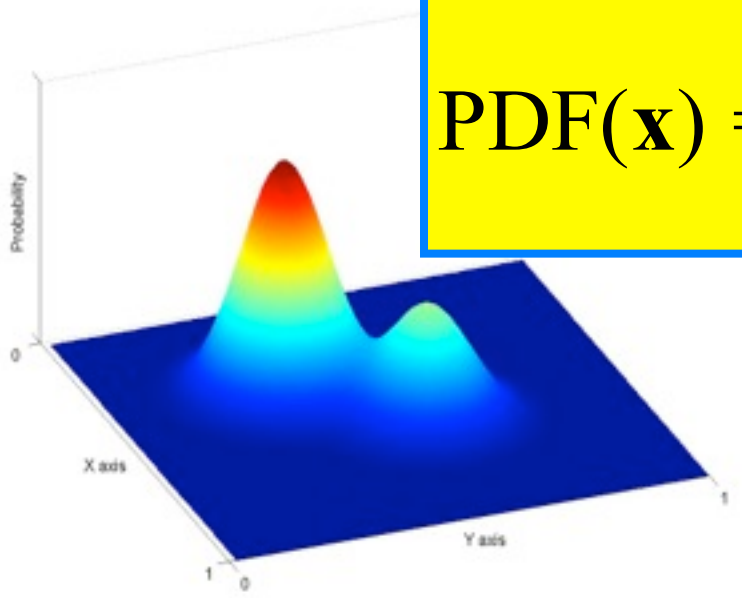
Assumed Underlying PDF

Real Data Samples

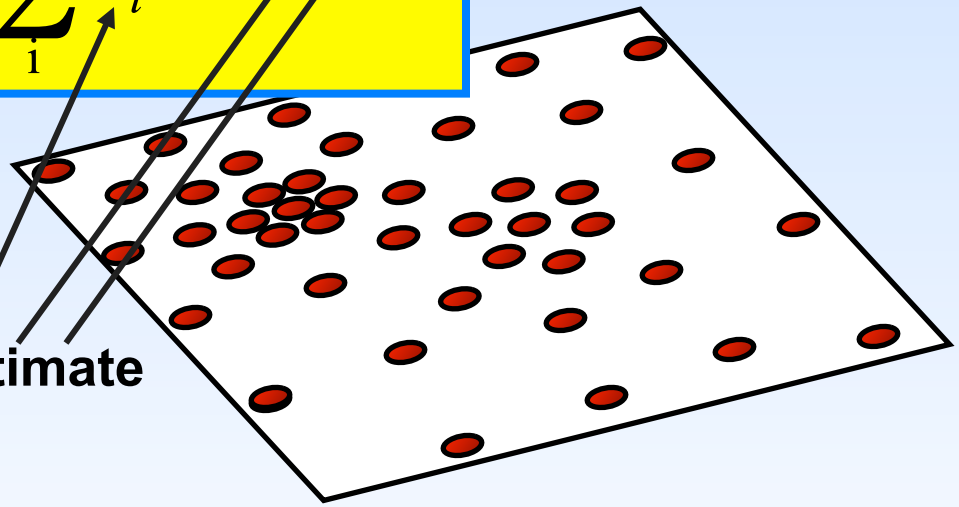
Parametric Density

Assumption : The data points are sampled from an underlying PDF

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



Estimate



Assumed Underlying PDF **Real Data Samples**

Kernel Density Estimation

Parzen Windows - Function Forms

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

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

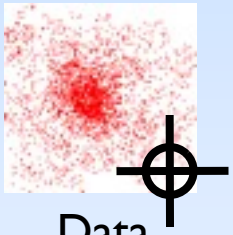
Data

Kernel Density Estimation

Parzen Windows - Function Forms

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

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



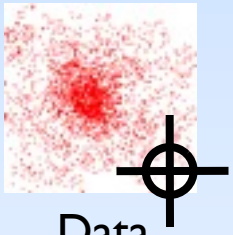
Data

Kernel Density Estimation

Parzen Windows - Function Forms

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

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



Data

In practice one uses the forms:

$$K(\mathbf{x}) = c\prod_{i=1}^{d} k(x_i)$$

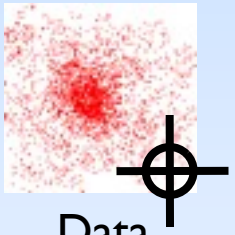
Same function on each dimension

Kernel Density Estimation

Parzen Windows - Function Forms

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

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



Data

In practice one uses the forms:

$$K(\mathbf{x}) = c\prod_{i=1}^{d} k(x_i)$$ or $$K(\mathbf{x}) = ck\left(\|\mathbf{x}\|\right)$$

Same function on each dimension

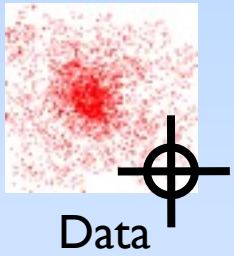
Function of vector length only

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 $x_1 \ldots x_n$



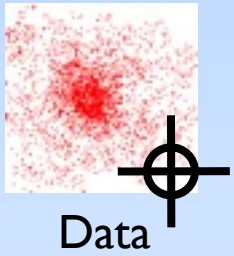
Data

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 $x_1 \ldots x_n$

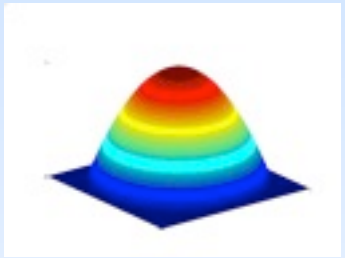


Data

Examples:

- Epanechnikov Kernel

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

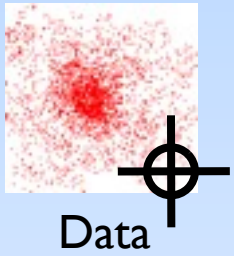


Kernel Density Estimation

Various Kernels

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

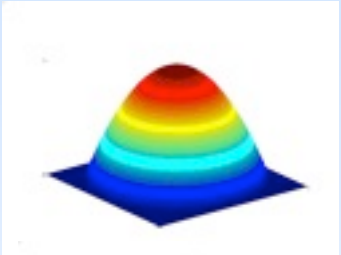
A function of some finite number of data points $x_1 \ldots x_n$



Examples:

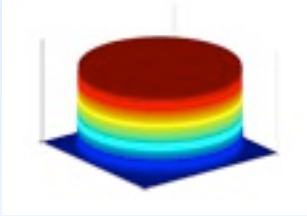
- Epanechnikov Kernel

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



- Uniform Kernel

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

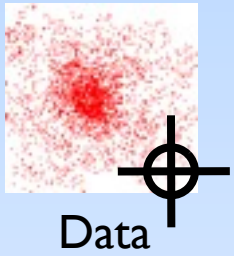


Kernel Density Estimation

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

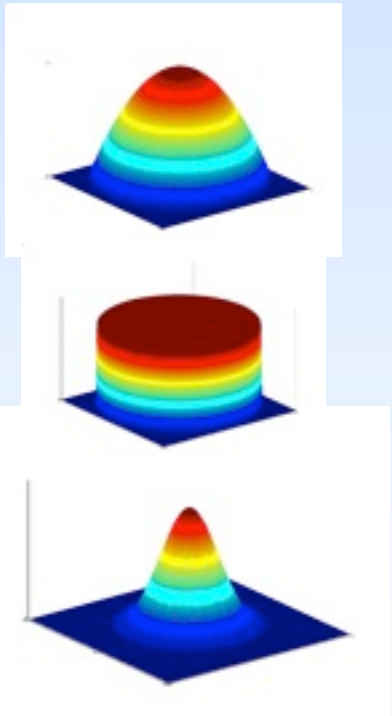
$$K_E(\mathbf{x}) = \begin{cases} c\left(1 - \|\mathbf{x}\|^2\right) & \|\mathbf{x}\| \le 1 \\ 0 & \text{otherwise} \end{cases}$$

- 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 Estimation

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

Kernel Density Estimation

Gradient

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

Kernel Density Estimation

Gradient

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

Kernel Density Estimation

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

Kernel Density Estimation

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:

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

Kernel Density Estimation

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

Kernel Density Estimation

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:

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

We get :

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

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

Kernel Density Estimation

Gradient

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

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

Kernel Density Estimation

Gradient

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

Computing The Mean Shift

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

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

Computing The Mean Shift

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

Computing The Mean Shift

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

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 !

Computing The Mean Shift

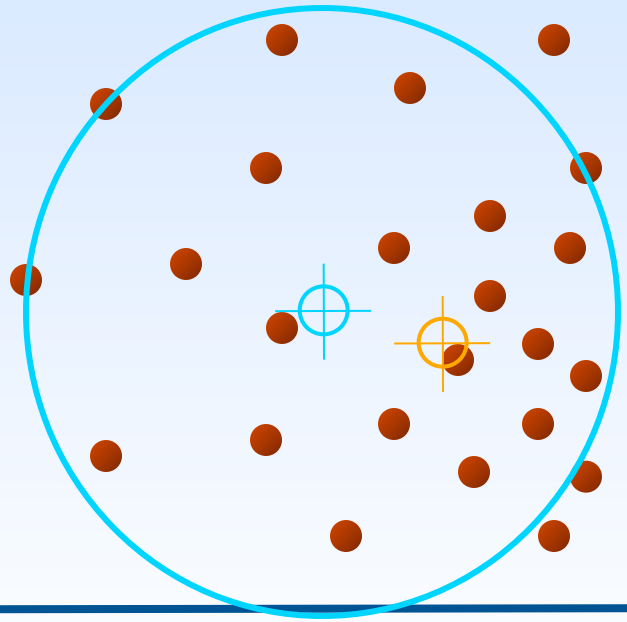
$$\nabla P(\mathbf{x}) = \frac{c}{n}\sum_{i=1}^{n}\nabla k_i = \frac{c}{n}\left[\sum_{i=1}^{n} g_i\right]\cdot\left[\frac{\sum_{i=1}^{n}\mathbf{x}_i g_i}{\sum_{i=1}^{n} g_i} - \mathbf{x}\right]$$

Yet another Kernel density estimation !

Computing The Mean Shift

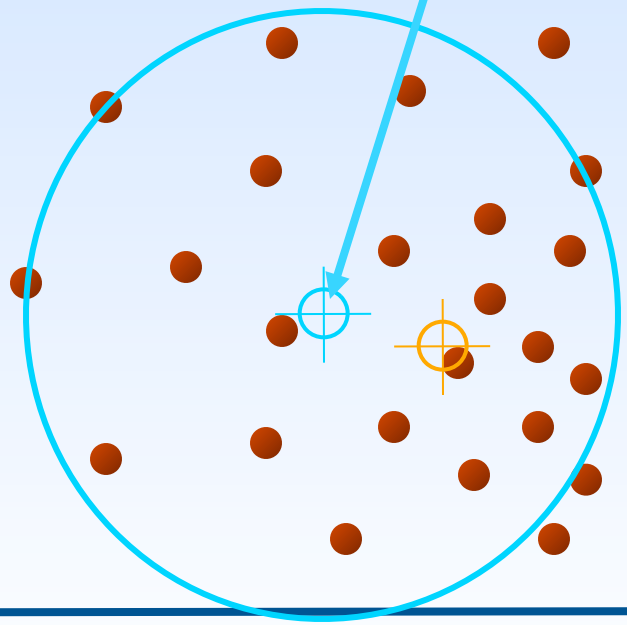
$$\nabla P(\mathbf{x}) = \frac{c}{n}\sum_{i=1}^{n}\nabla k_i = \frac{c}{n}\left[\sum_{i=1}^{n} g_i\right]\cdot\left[\frac{\sum_{i=1}^{n}\mathbf{x}_i g_i}{\sum_{i=1}^{n} g_i} - \mathbf{x}\right]$$

Yet another Kernel density estimation !



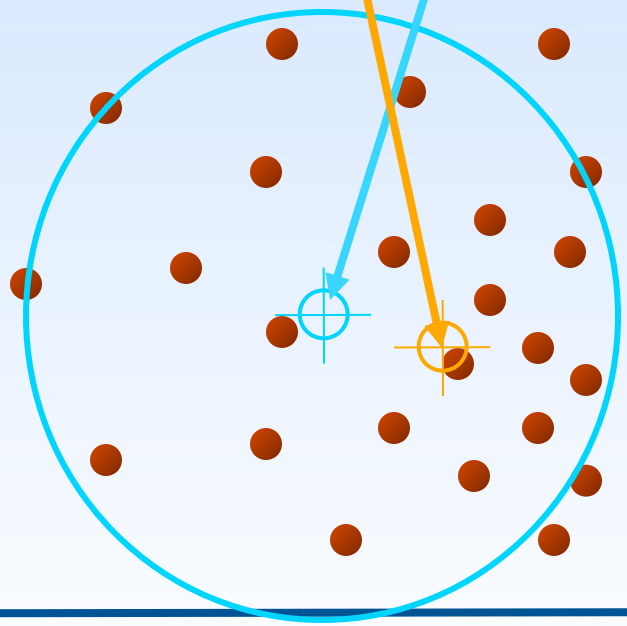
Computing The Mean Shift

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



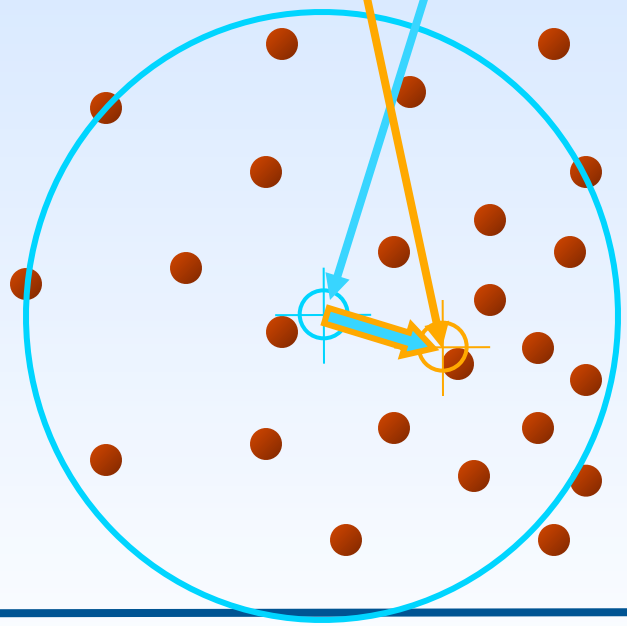
Computing The Mean Shift

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



Computing The Mean Shift

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



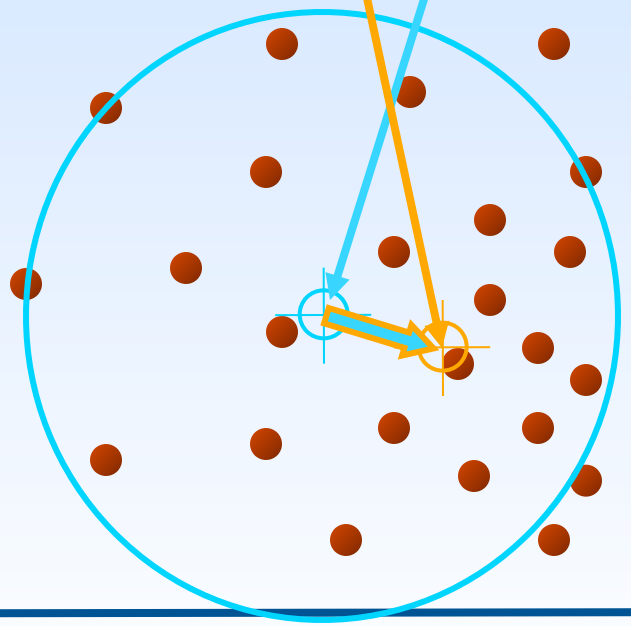
Computing The Mean Shift

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

Simple Mean Shift procedure:

- Compute mean shift vector

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



Computing The Mean Shift

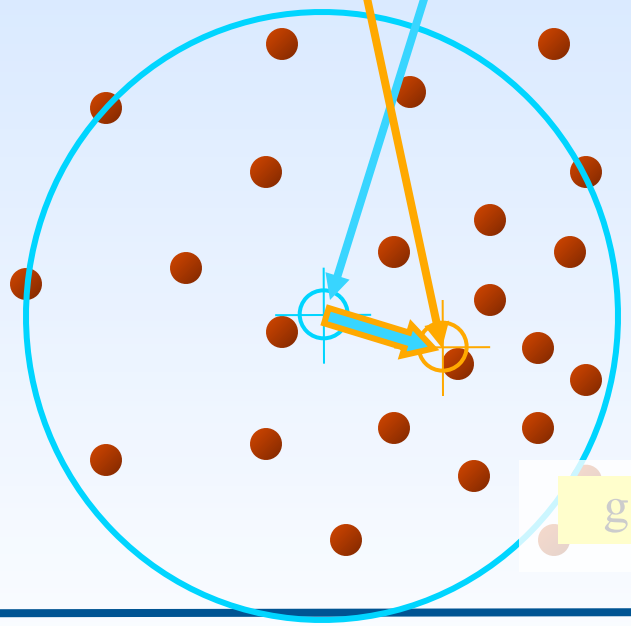
$$\nabla P(\mathbf{x}) = \frac{c}{n}\sum_{i=1}^{n}\nabla k_i = \frac{c}{n}\left[\sum_{i=1}^{n} g_i\right]\left[\frac{\sum_{i=1}^{n}\mathbf{x}_i g_i}{\sum_{i=1}^{n} g_i} - \mathbf{x}\right]$$

Simple Mean Shift procedure:

- Compute mean shift vector

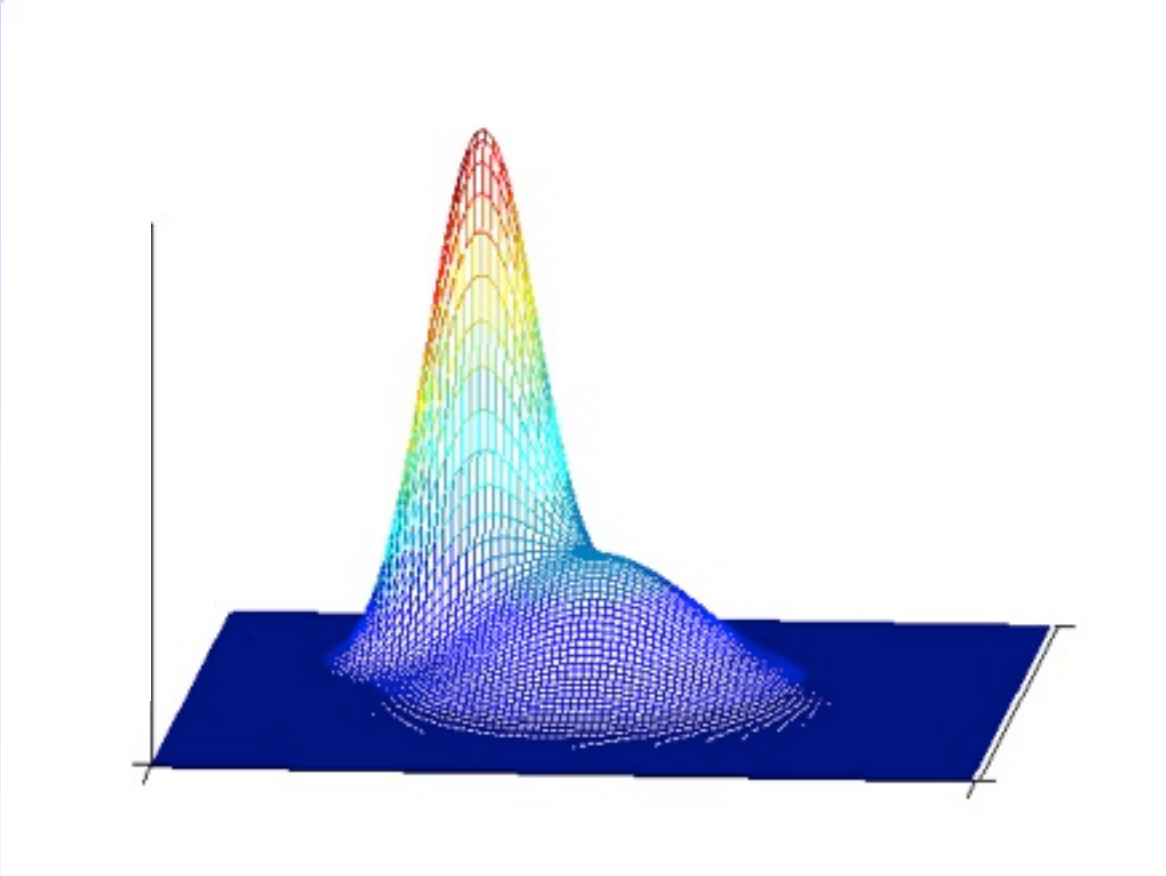
$$\mathbf{m}(\mathbf{x}) = \left[\frac{\sum_{i=1}^{n}\mathbf{x}_i g\left(\left\|\frac{\mathbf{x}-\mathbf{x}_i}{h}\right\|^2\right)}{\sum_{i=1}^{n} g\left(\left\|\frac{\mathbf{x}-\mathbf{x}_i}{h}\right\|^2\right)} - \mathbf{x}\right]$$

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

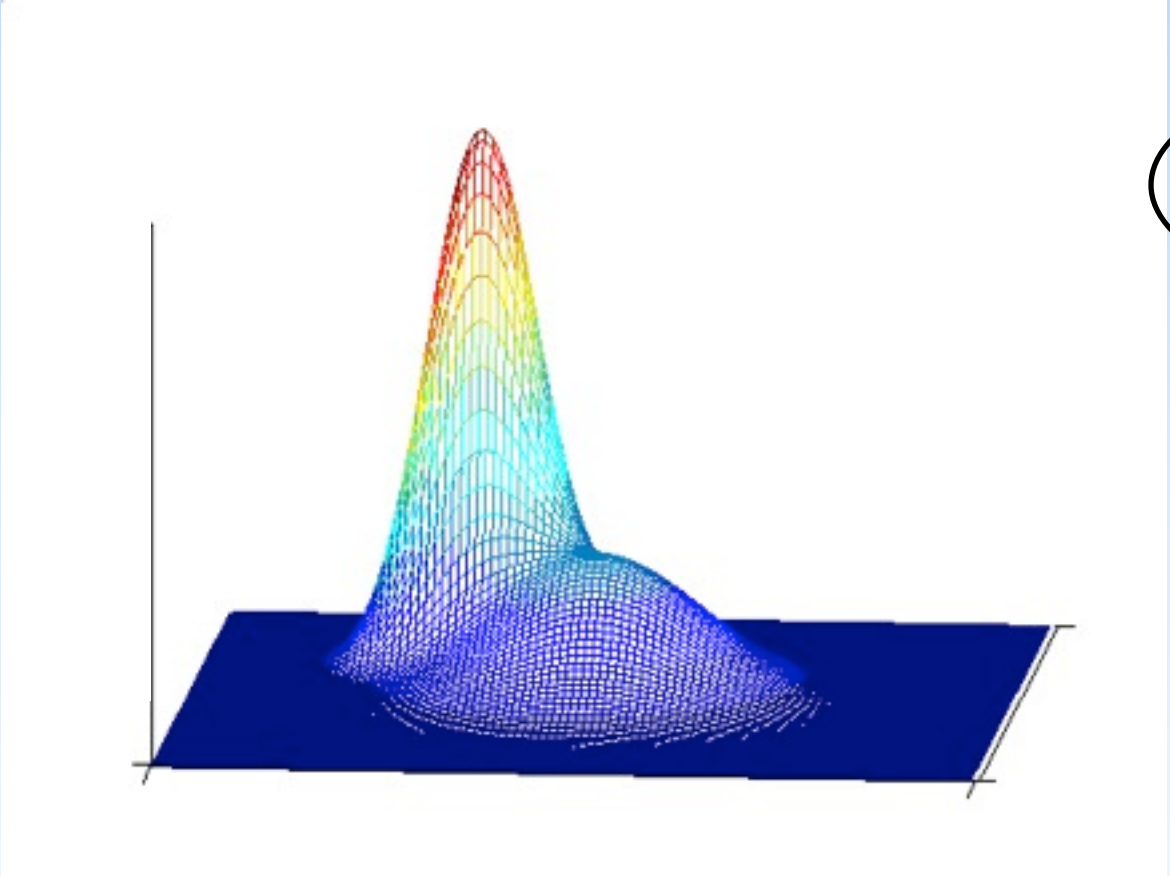


$$g(\mathbf{x}) = -k'(\mathbf{x})$$

Mean Shift Mode Detection

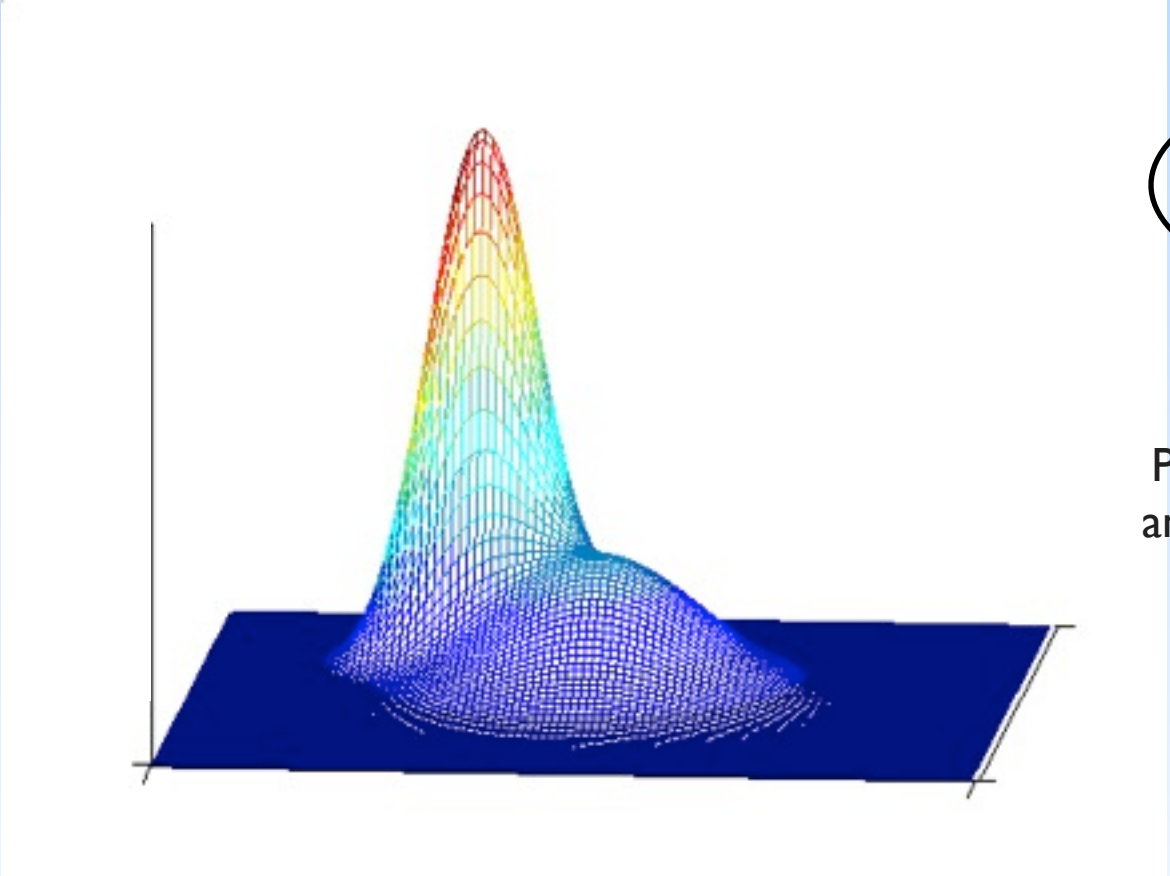


Mean Shift Mode Detection



What happens if we reach a saddle point ?

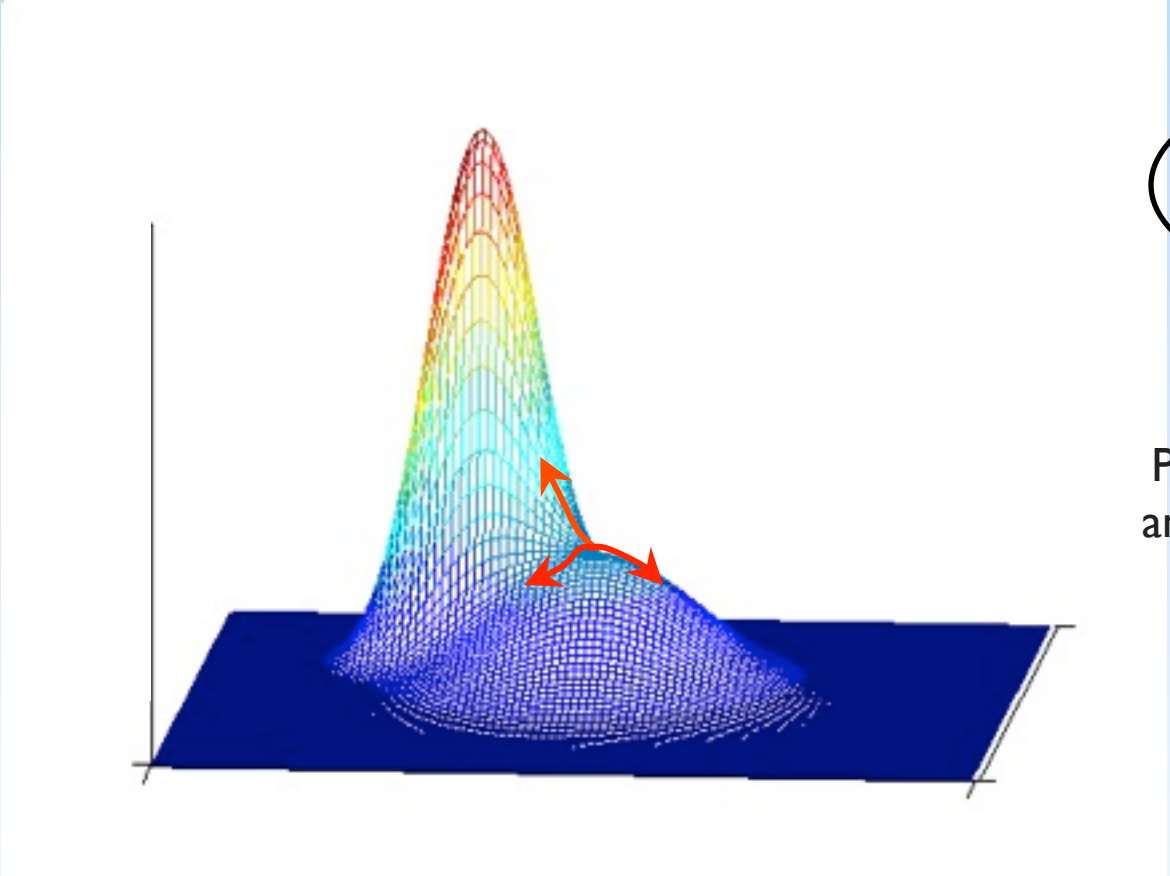
Mean Shift Mode Detection



What happens if we reach a saddle point ?

Perturb the mode position and check if we return back

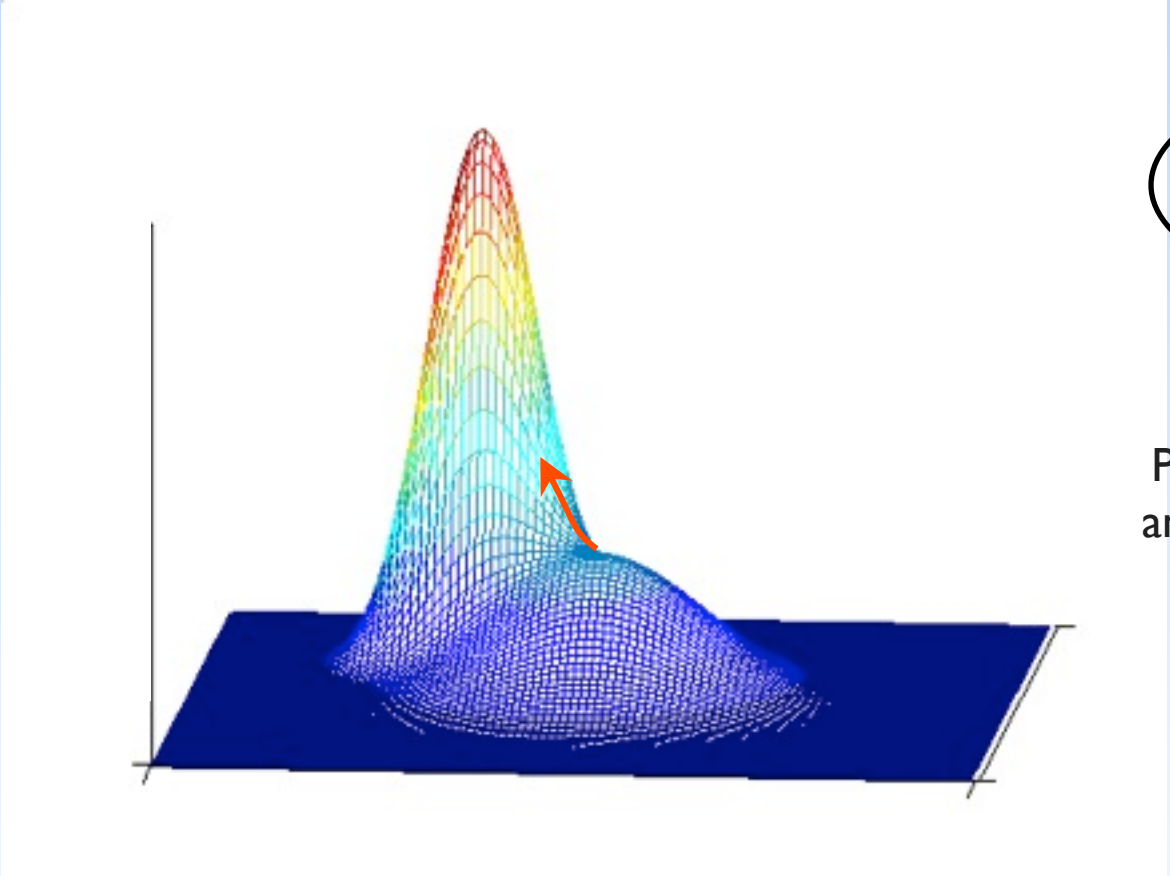
Mean Shift Mode Detection



What happens if we reach a saddle point ?

Perturb the mode position and check if we return back

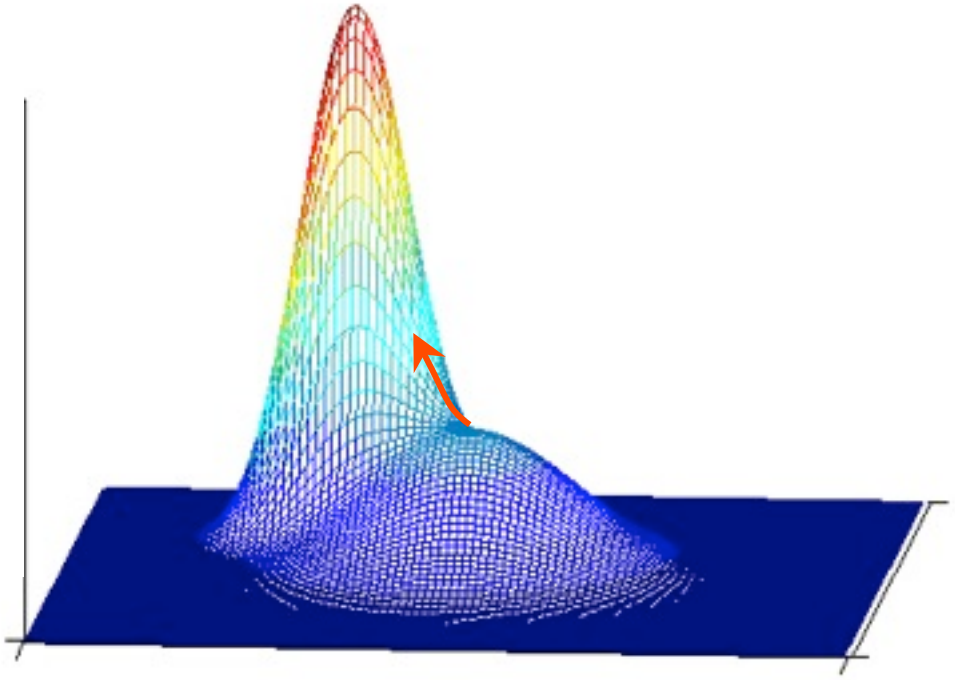
Mean Shift Mode Detection



What happens if we reach a saddle point ?

Perturb the mode position and check if we return back

Mean Shift Mode Detection

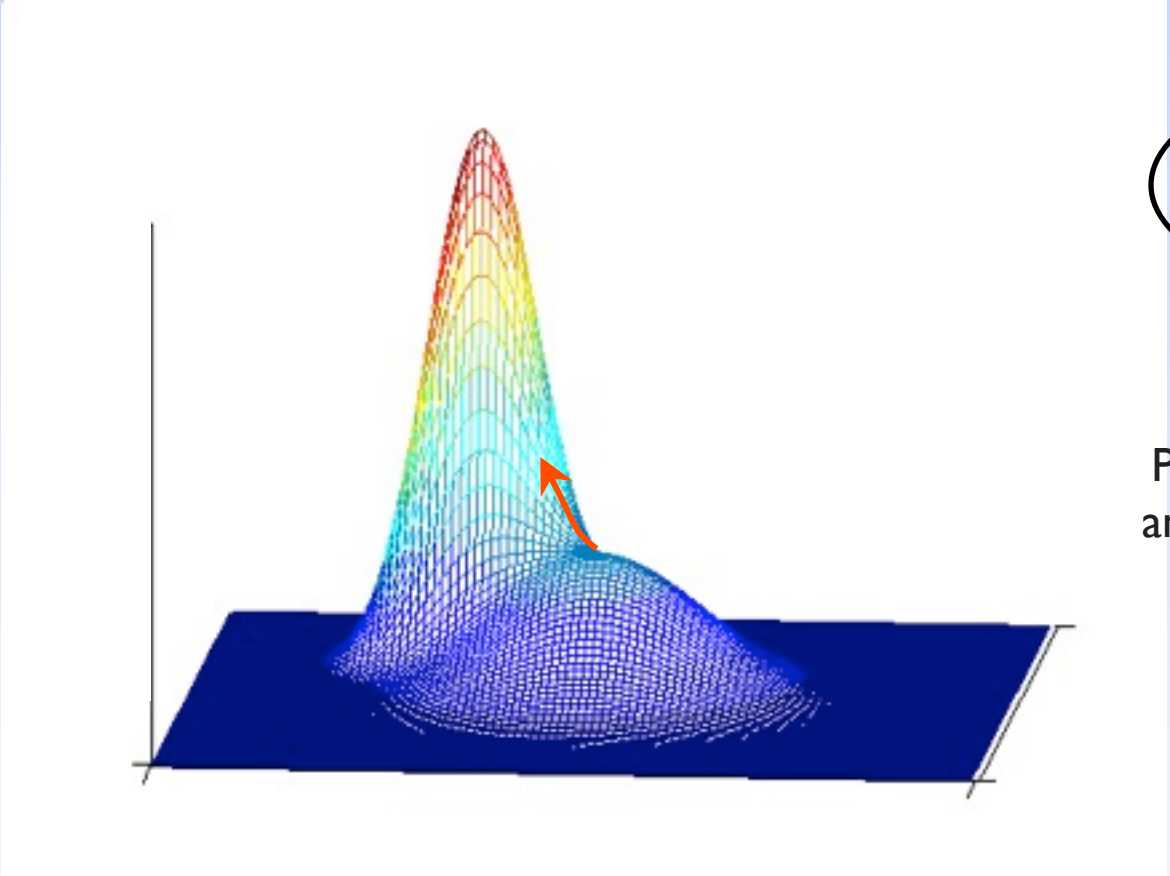


What happens if we reach a saddle point ?

Perturb the mode position and check if we return back

Updated Mean Shift Procedure:

Mean Shift Mode Detection



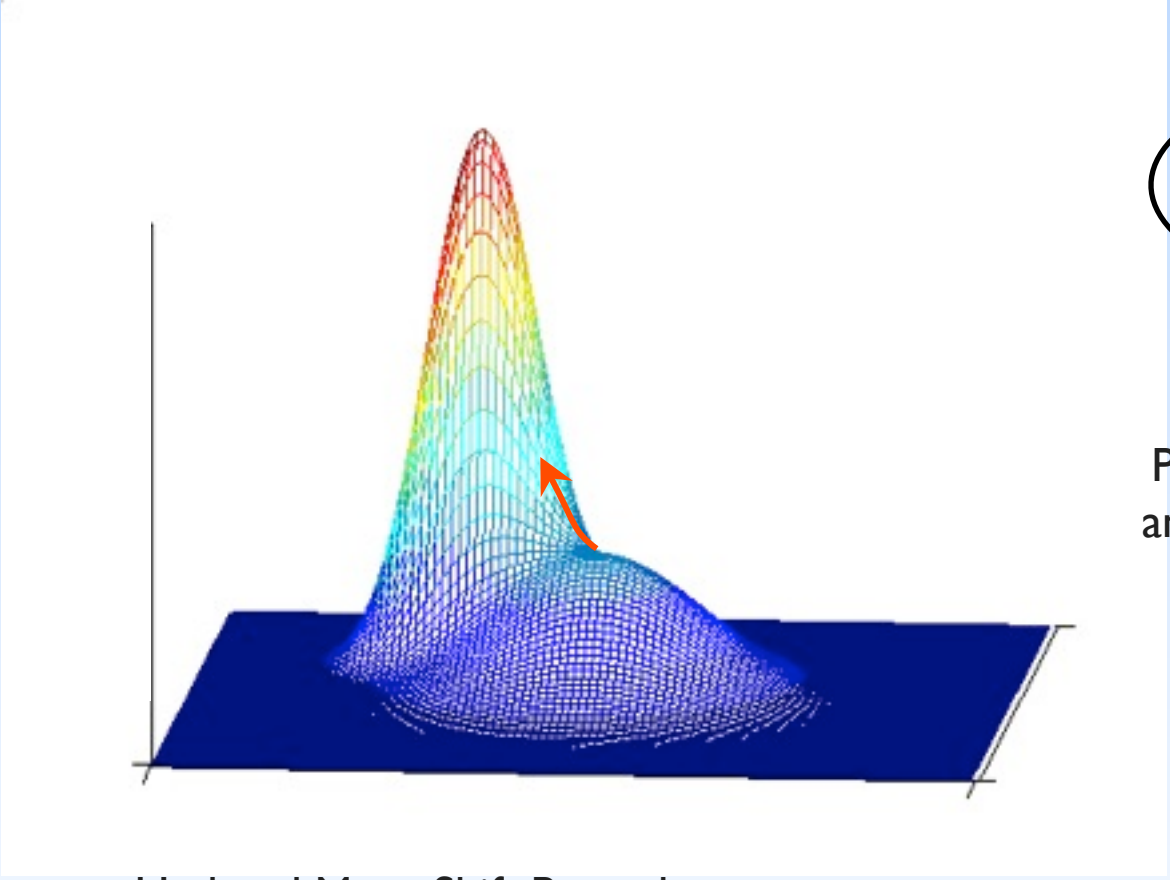
What happens if we reach a saddle point ?

Perturb the mode position and check if we return back

Updated Mean Shift Procedure:

- Find all modes using the Simple Mean Shift Procedure

Mean Shift Mode Detection



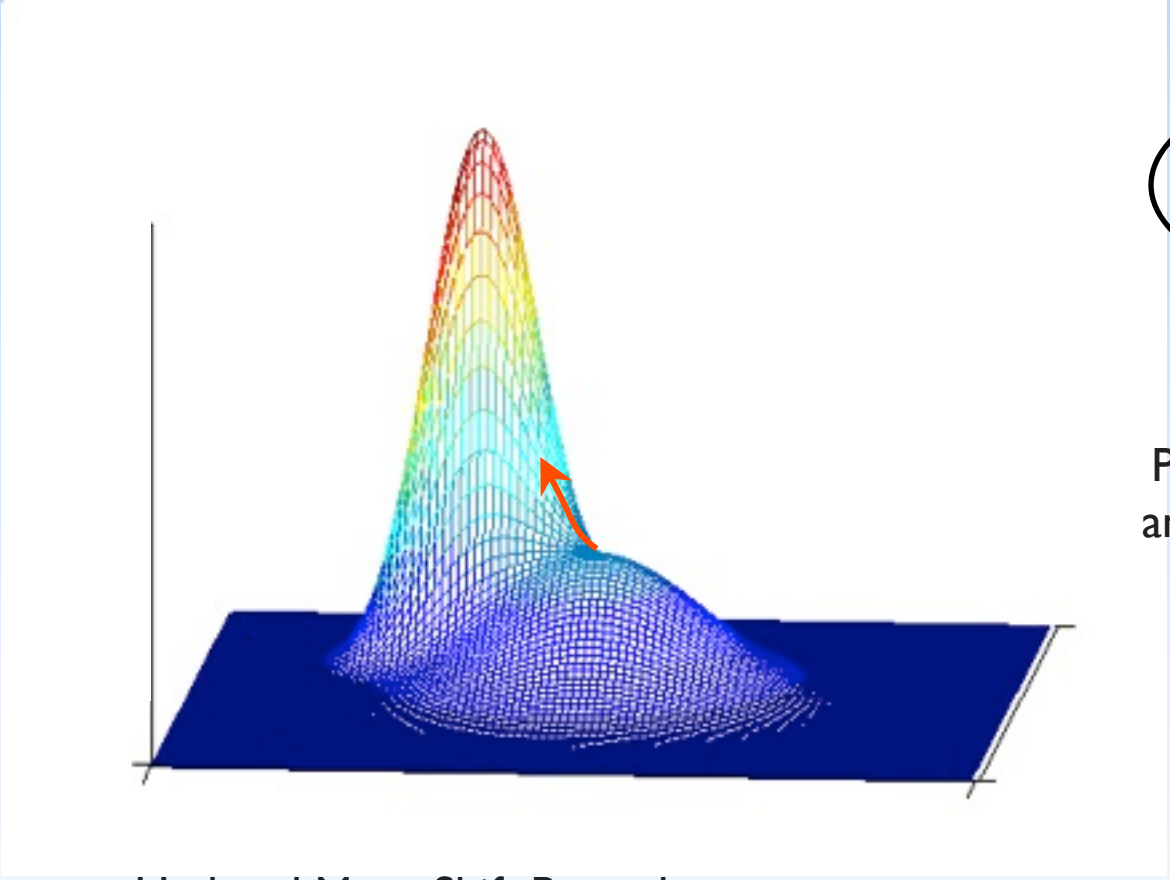
What happens if we reach a saddle point ?

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)

Mean Shift Mode Detection



What happens if we reach a saddle point ?

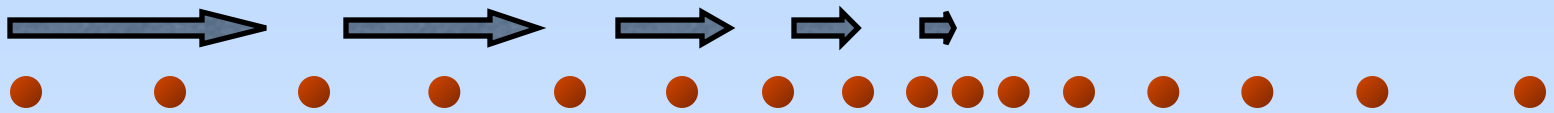
Perturb the mode position and check if we return back

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

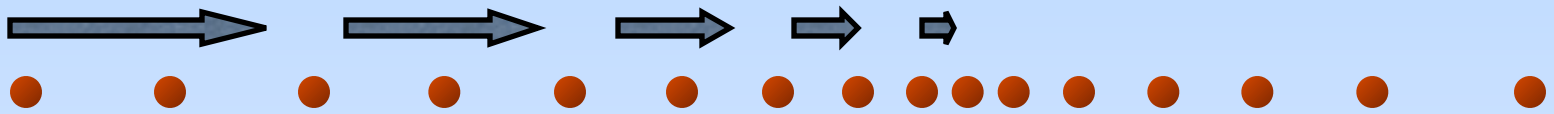
Mean Shift Properties



- Automatic convergence speed – the mean shift vector size depends on the gradient itself.

- Near maxima, the steps are small and refined

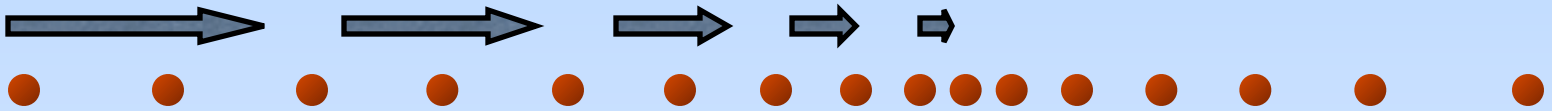
Mean Shift Properties



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

Adaptive Gradient Ascent

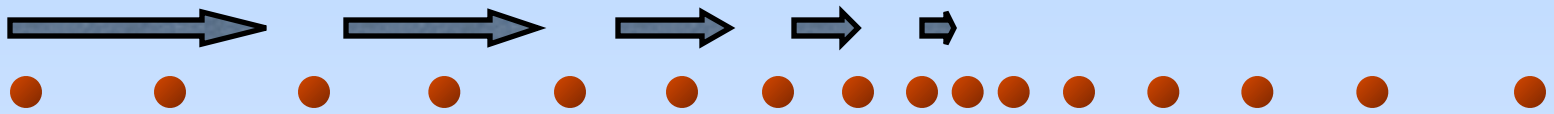
Mean Shift Properties



- Automatic convergence speed – the mean shift vector size depends on the gradient itself.
- Near maxima, the steps are small and refined
- Convergence is guaranteed for infinitesimal steps only → infinitely convergent, (therefore set a lower bound)

Adaptive Gradient Ascent

Mean Shift Properties

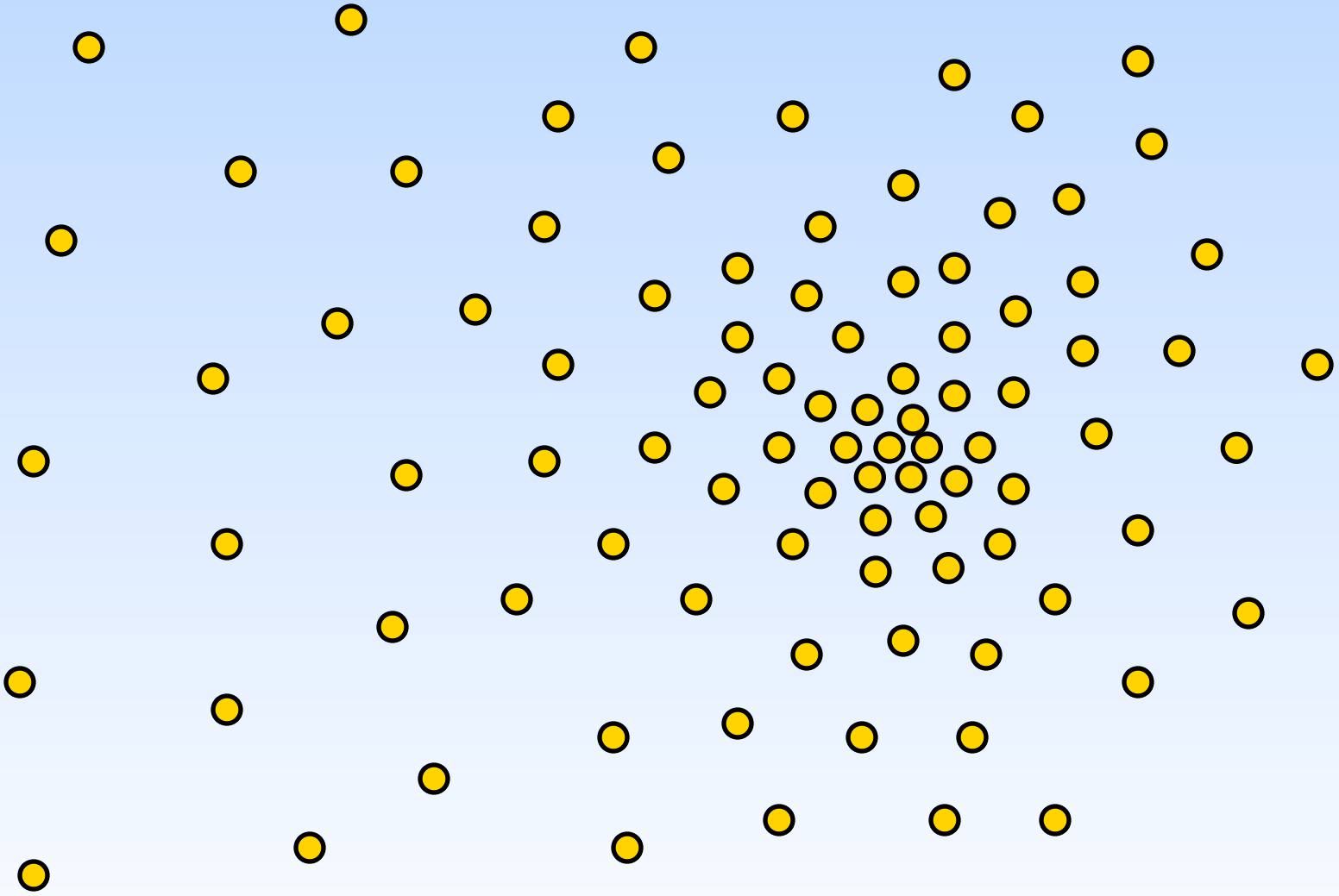


- Automatic convergence speed – the mean shift vector size depends on the gradient itself.
- Near maxima, the steps are small and refined

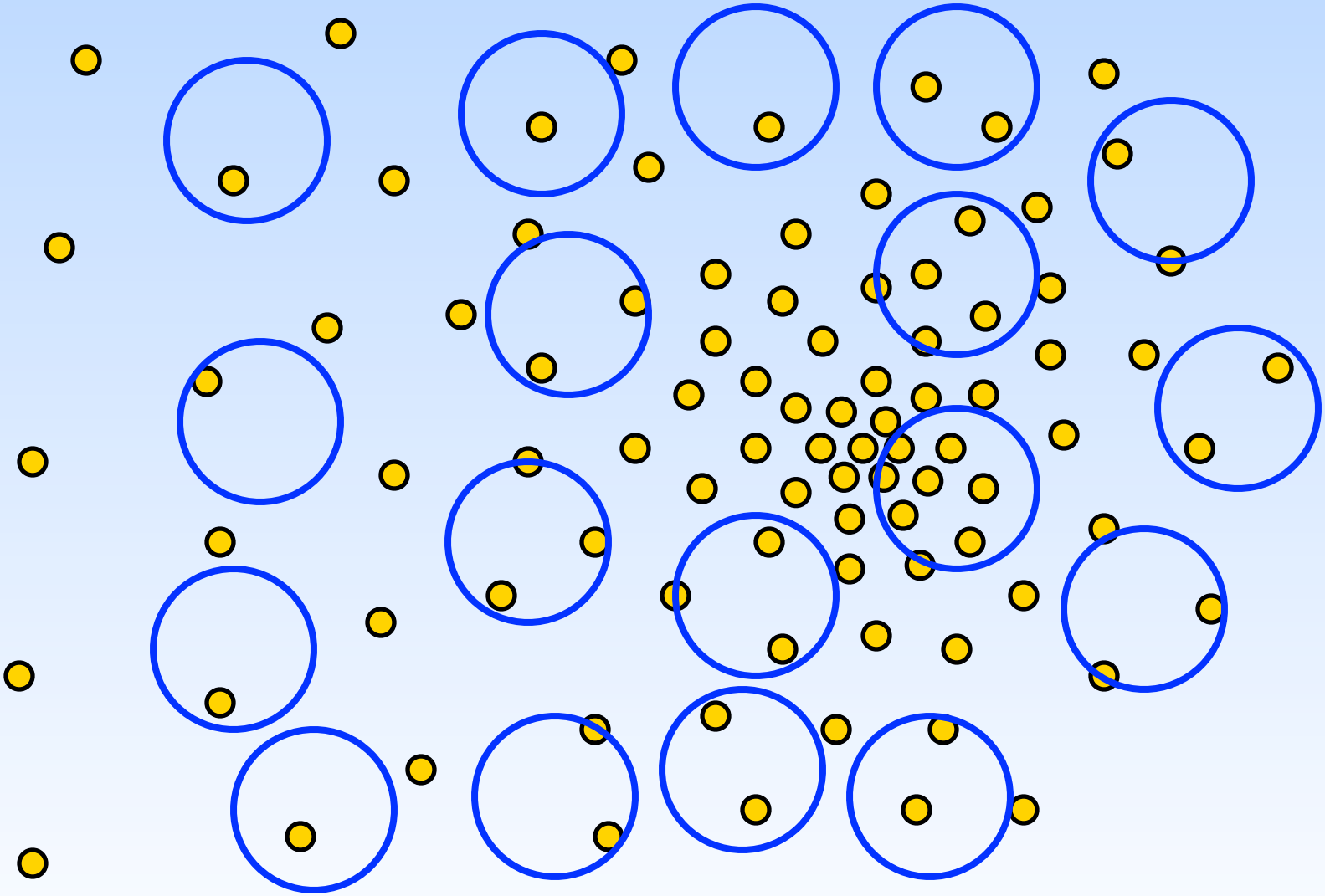
Adaptive Gradient Ascent

- 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

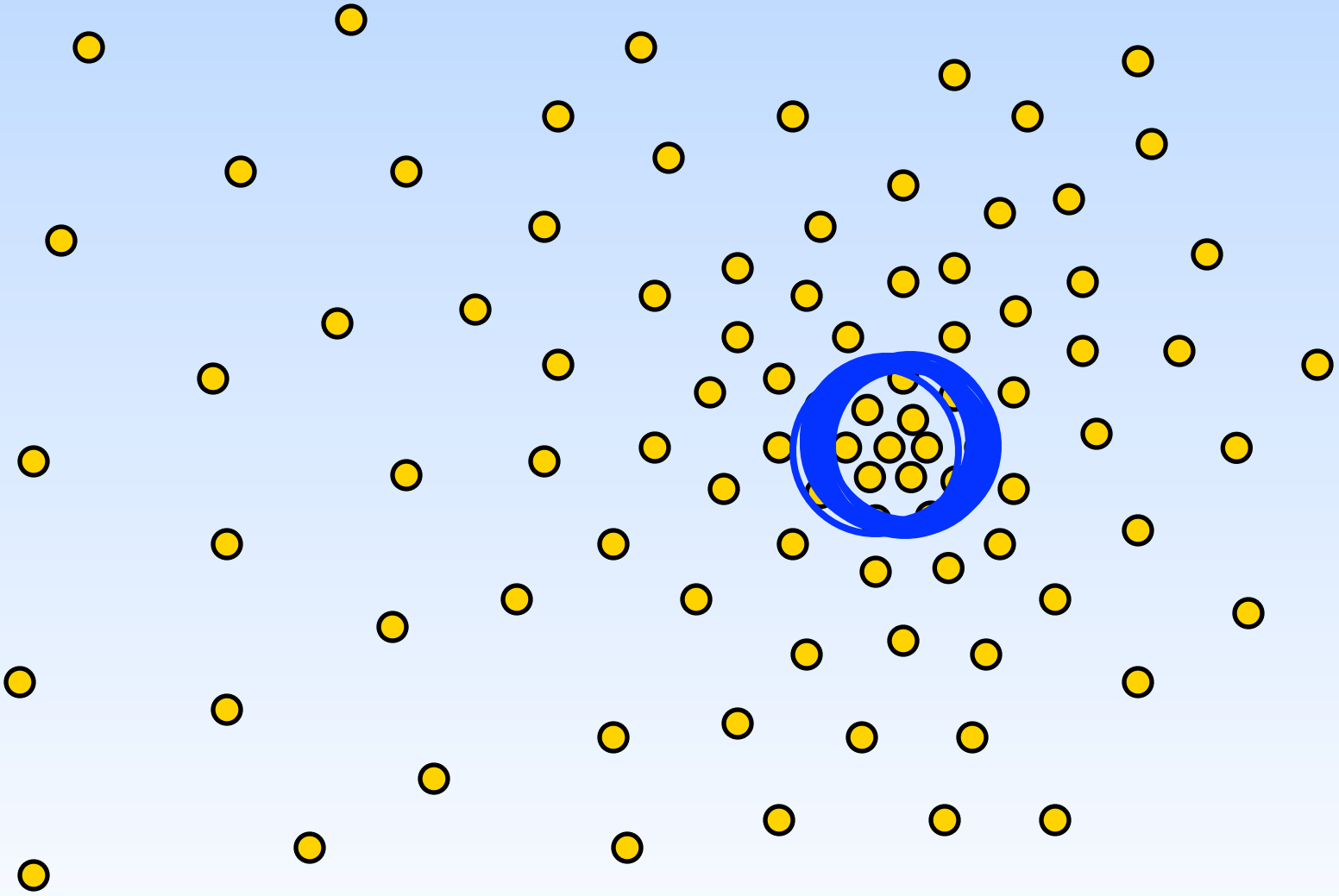


Real Modality Analysis



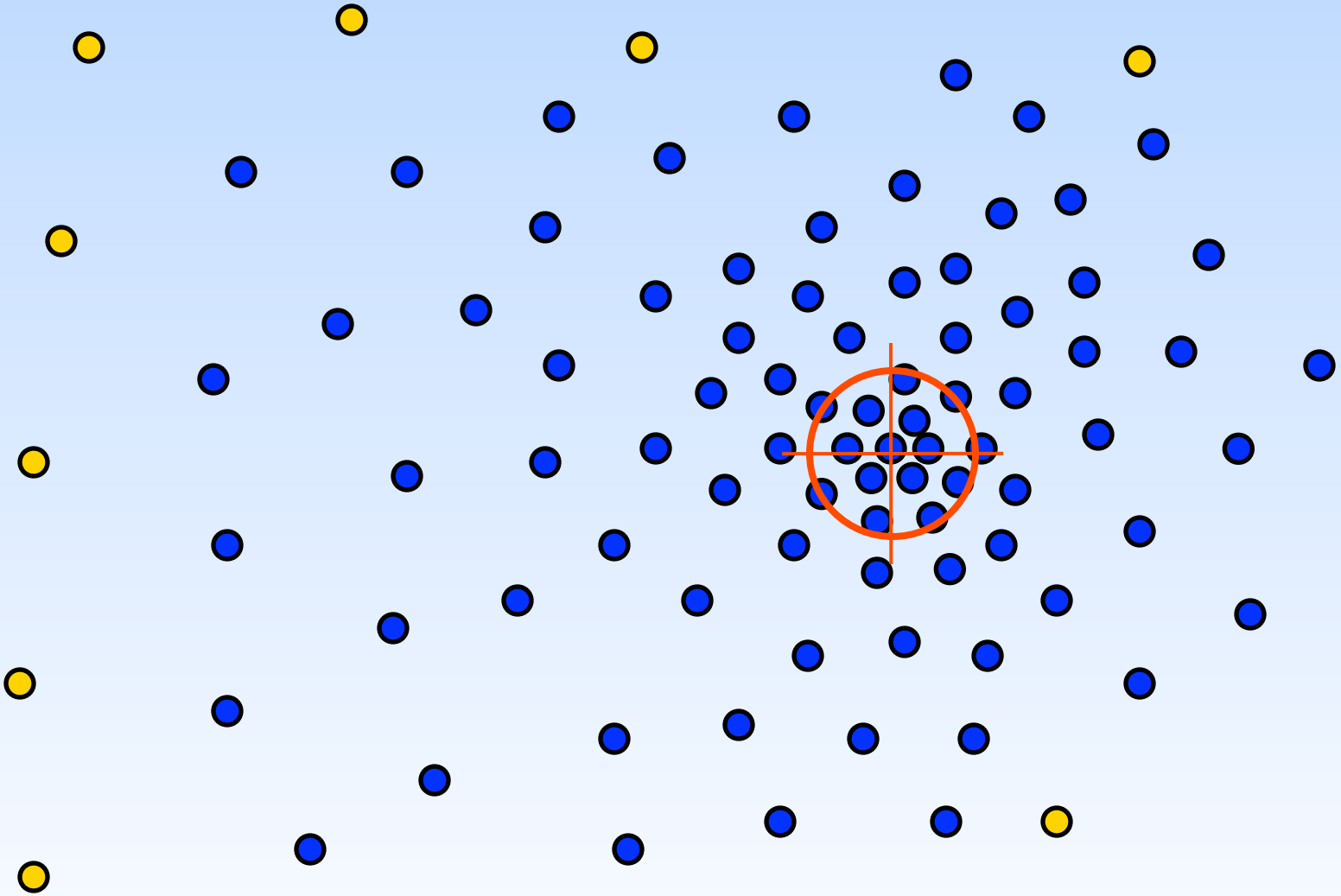
Run the procedure in parallel

Real Modality Analysis



Run the procedure in parallel

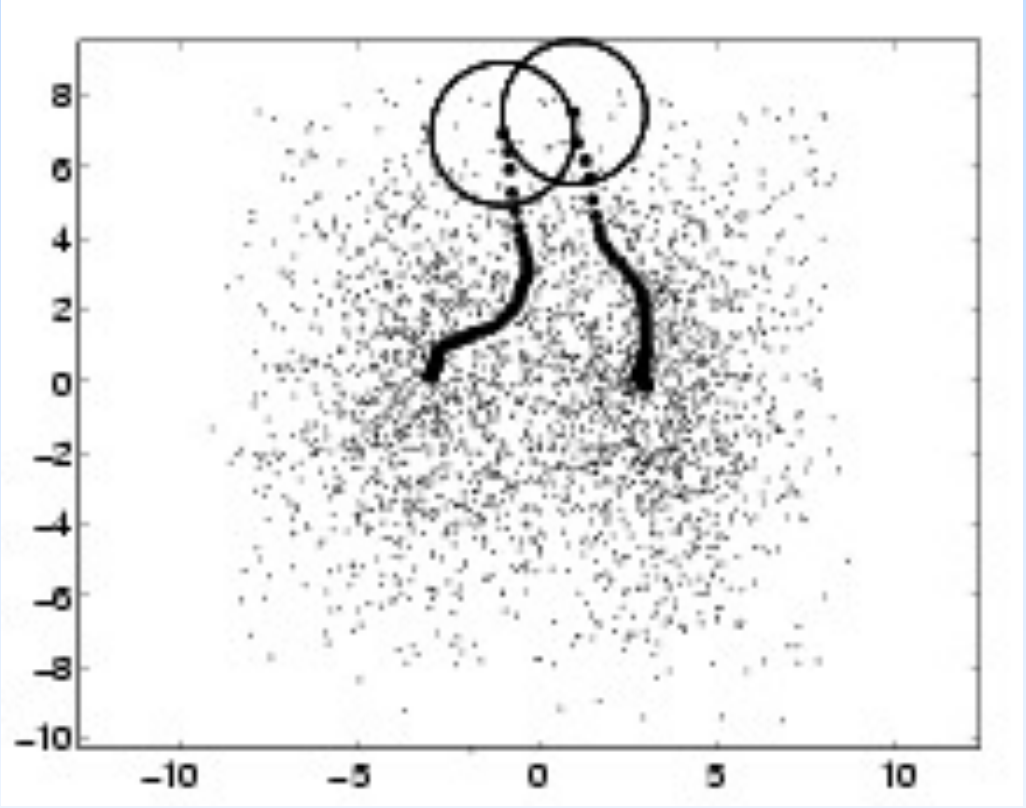
Real Modality Analysis



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

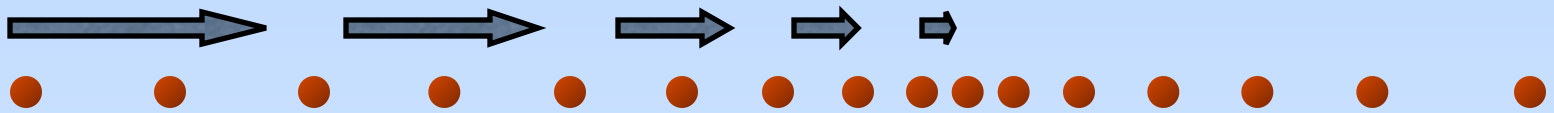
Real Modality Analysis

An example

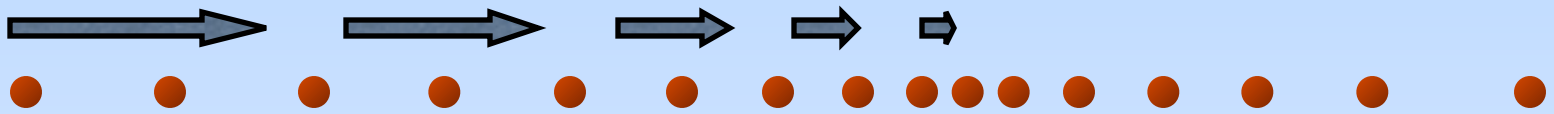


Window tracks signify the steepest ascent directions

Mean Shift Strengths & Weaknesses

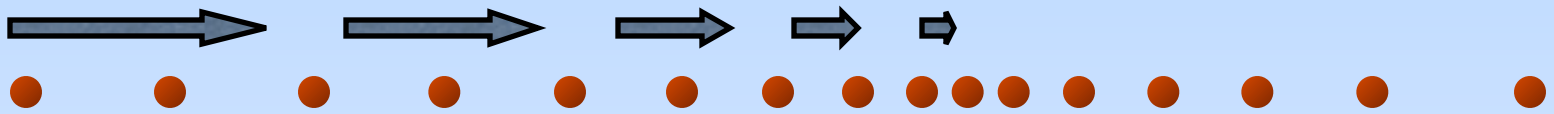


Mean Shift Strengths & Weaknesses



Strengths :

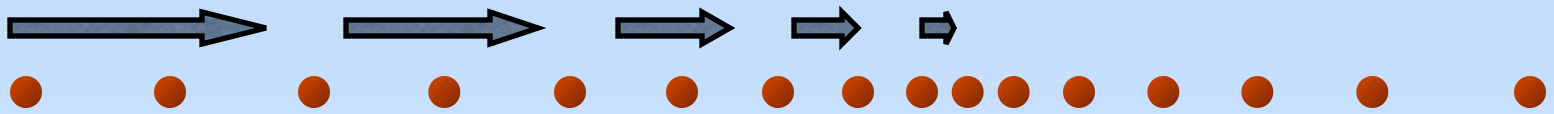
Mean Shift Strengths & Weaknesses



Strengths :

- Application independent tool

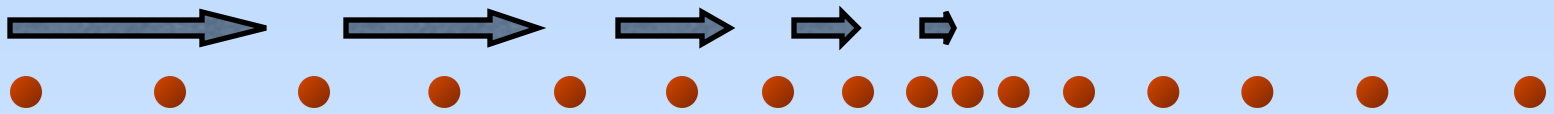
Mean Shift Strengths & Weaknesses



Strengths :

- Application independent tool
- Suitable for real data analysis

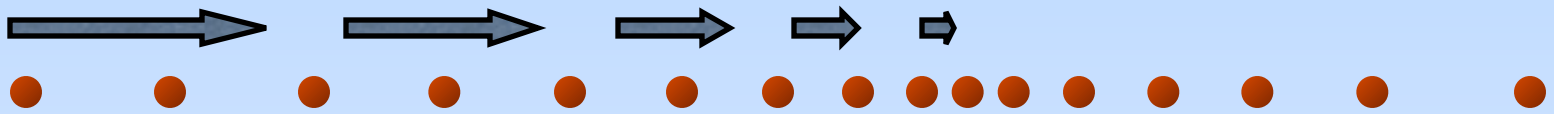
Mean Shift Strengths & Weaknesses



Strengths :

- Application independent tool
- Suitable for real data analysis
- Does not assume any prior shape
  (e.g. elliptical) on data clusters

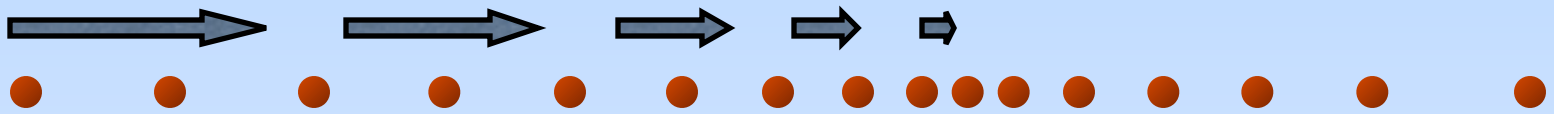
Mean Shift Strengths & Weaknesses



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

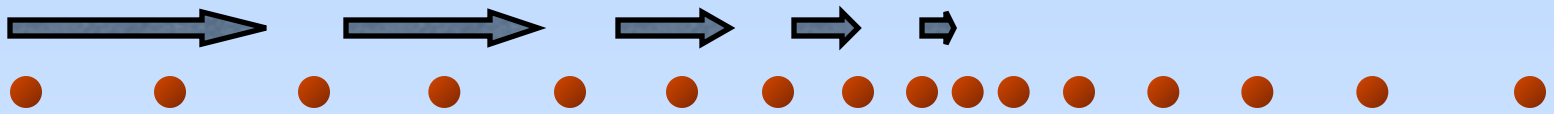
Mean Shift Strengths & Weaknesses



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

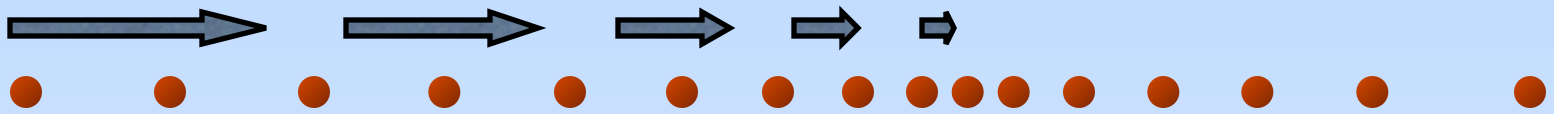
Mean Shift Strengths & Weaknesses



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

Mean Shift Strengths & Weaknesses

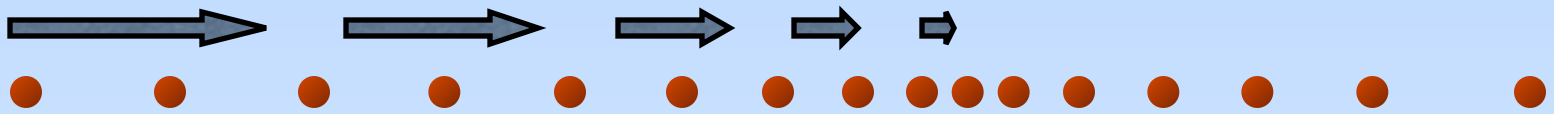


Strengths :

- 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 :

Mean Shift Strengths & Weaknesses



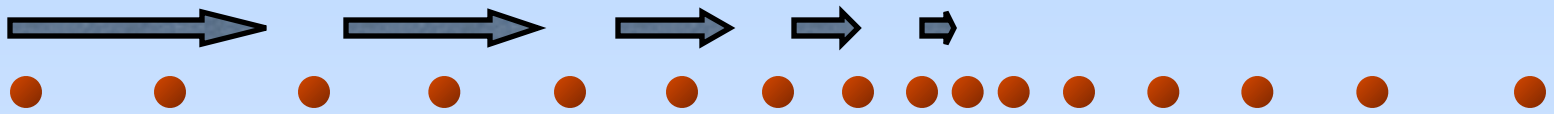
Strengths :

- 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

Mean Shift Strengths & Weaknesses



Strengths :

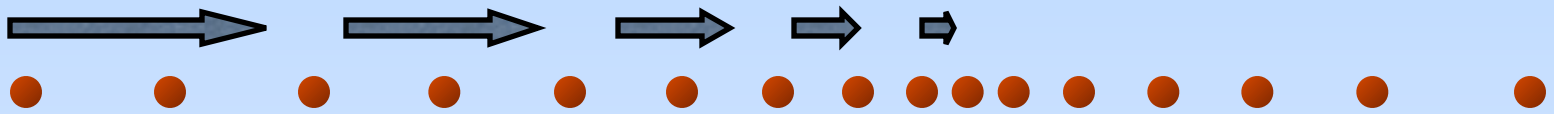
- Application independent tool
- Suitable for real data analysis
- Does not assume any prior shape (e.g. elliptical) on data clusters
- Can handle arbitrary feature spaces
- Only ONE parameter to choose
- *h* (window size) has a physical meaning, unlike K-Means

Weaknesses :

- The window size (bandwidth selection) is not trivial

- Inappropriate window size can cause modes to be merged, or generate additional "shallow" modes

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

Mean Shift Applications

Clustering

Cluster : All data points in the ***attraction basin*** of a mode

Mean Shift : A robust Approach Toward Feature Space Analysis, by Comaniciu, Meer

Clustering

Cluster : All data points in the ***attraction basin*** of a mode

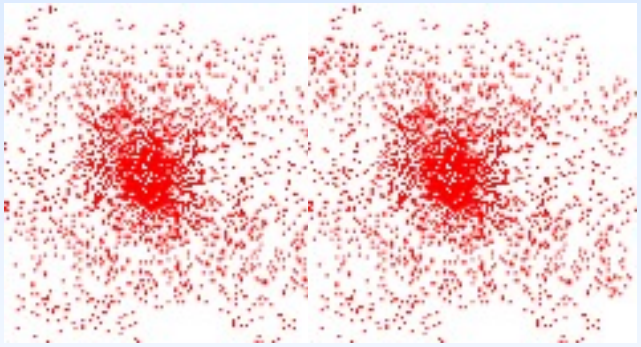
Attraction basin : the region for which all trajectories lead to the same mode

Mean Shift : A robust Approach Toward Feature Space Analysis, by Comaniciu, Meer

Clustering

Cluster : All data points in the ***attraction basin*** of a mode

Attraction basin : the region for which all trajectories lead to the same mode

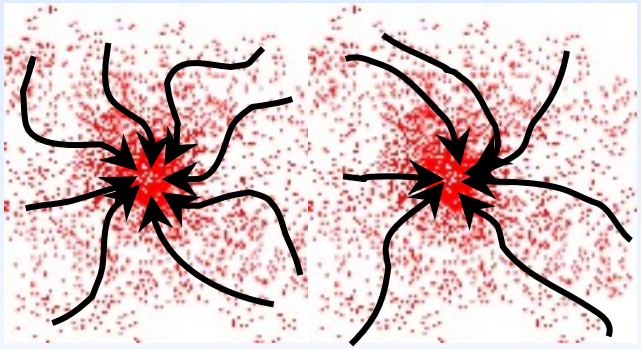


Mean Shift : A robust Approach Toward Feature Space Analysis, by Comaniciu, Meer

Clustering

Cluster : All data points in the ***attraction basin*** of a mode

Attraction basin : the region for which all trajectories lead to the same mode

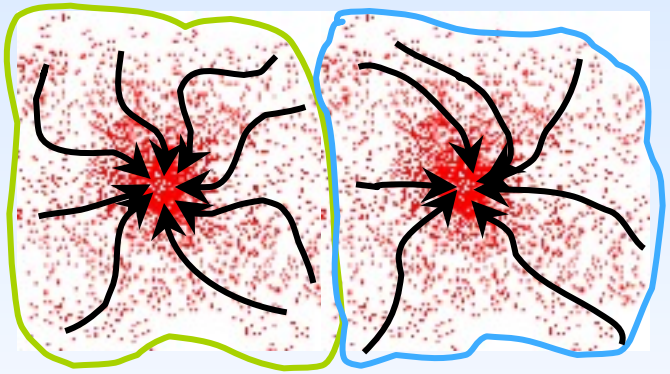


Mean Shift : A robust Approach Toward Feature Space Analysis, by Comaniciu, Meer

Clustering

Cluster : All data points in the ***attraction basin*** of a mode

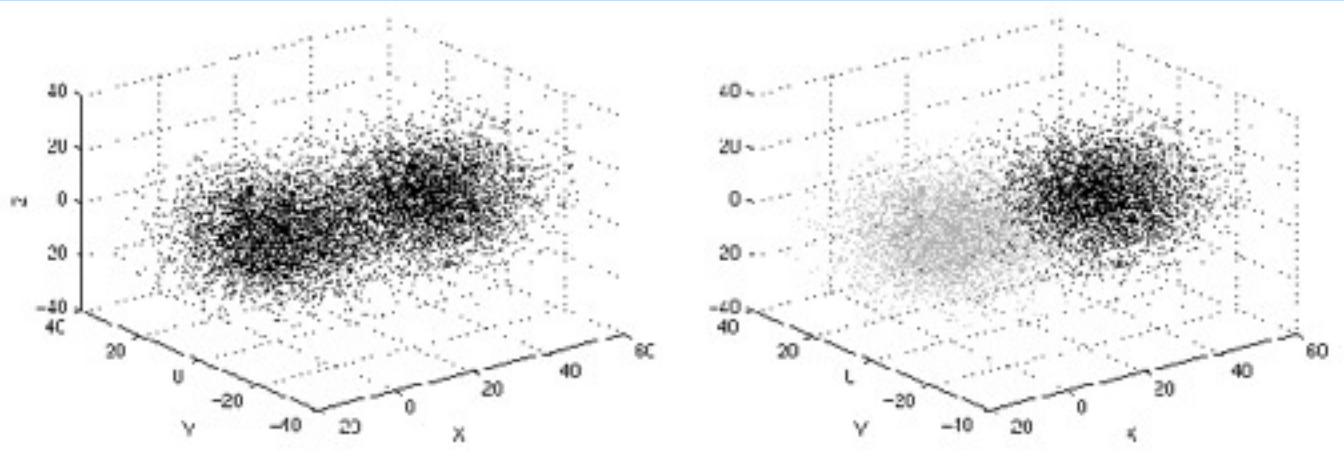
Attraction basin : the region for which all trajectories lead to the same mode



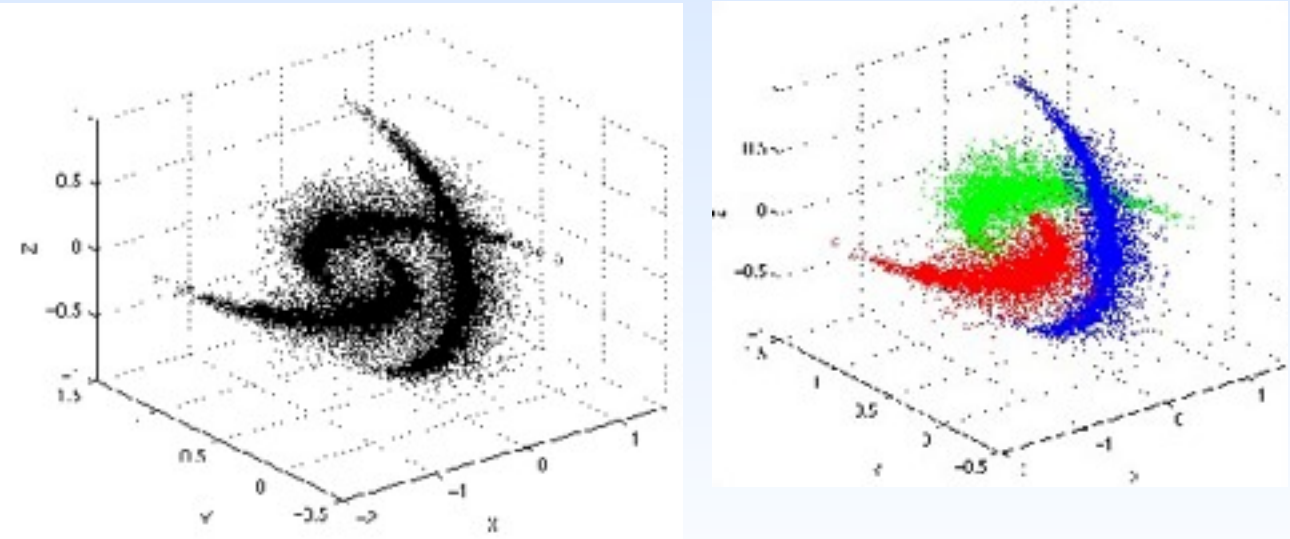
Mean Shift : A robust Approach Toward Feature Space Analysis, by Comaniciu, Meer

Clustering

Synthetic Examples

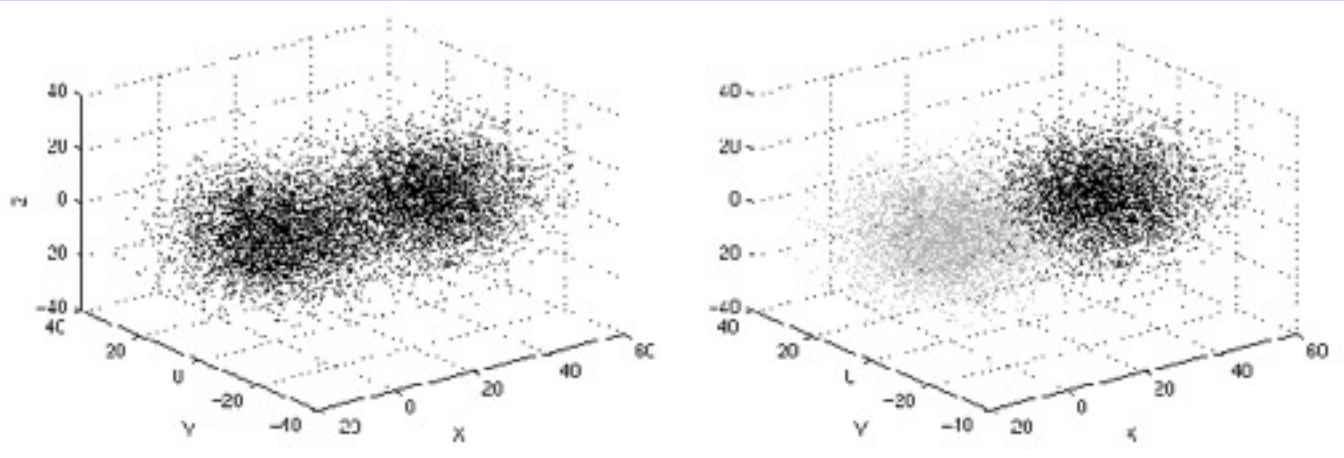


Simple Modal Structures

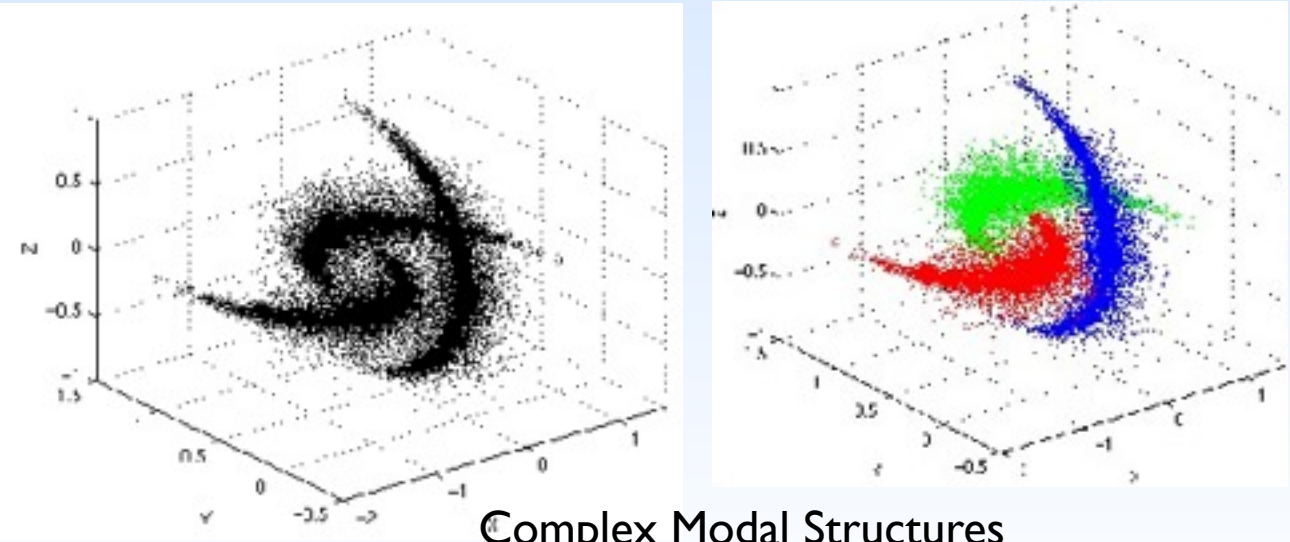


Clustering

Synthetic Examples



Simple Modal Structures

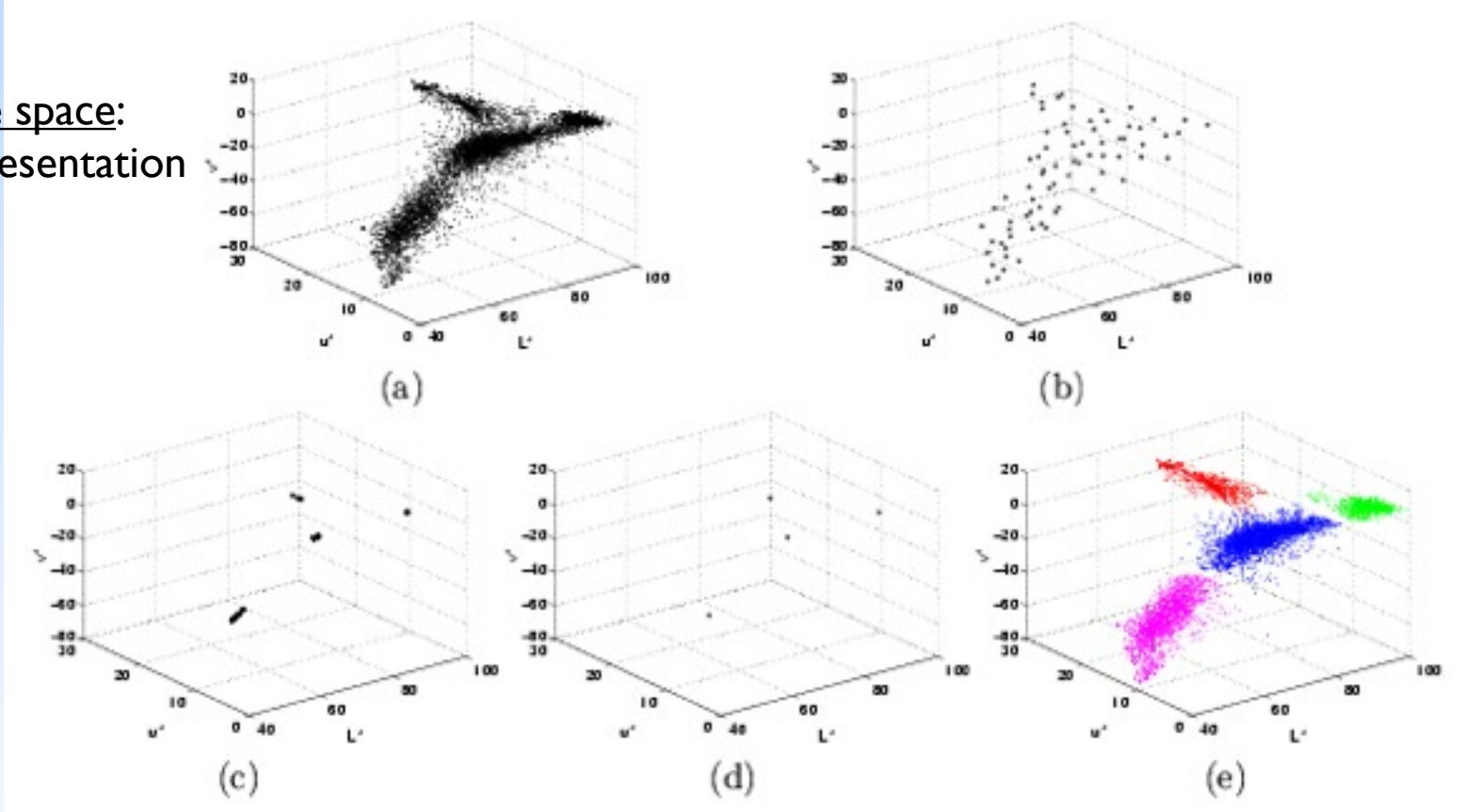


Complex Modal Structures

Clustering

Real Example

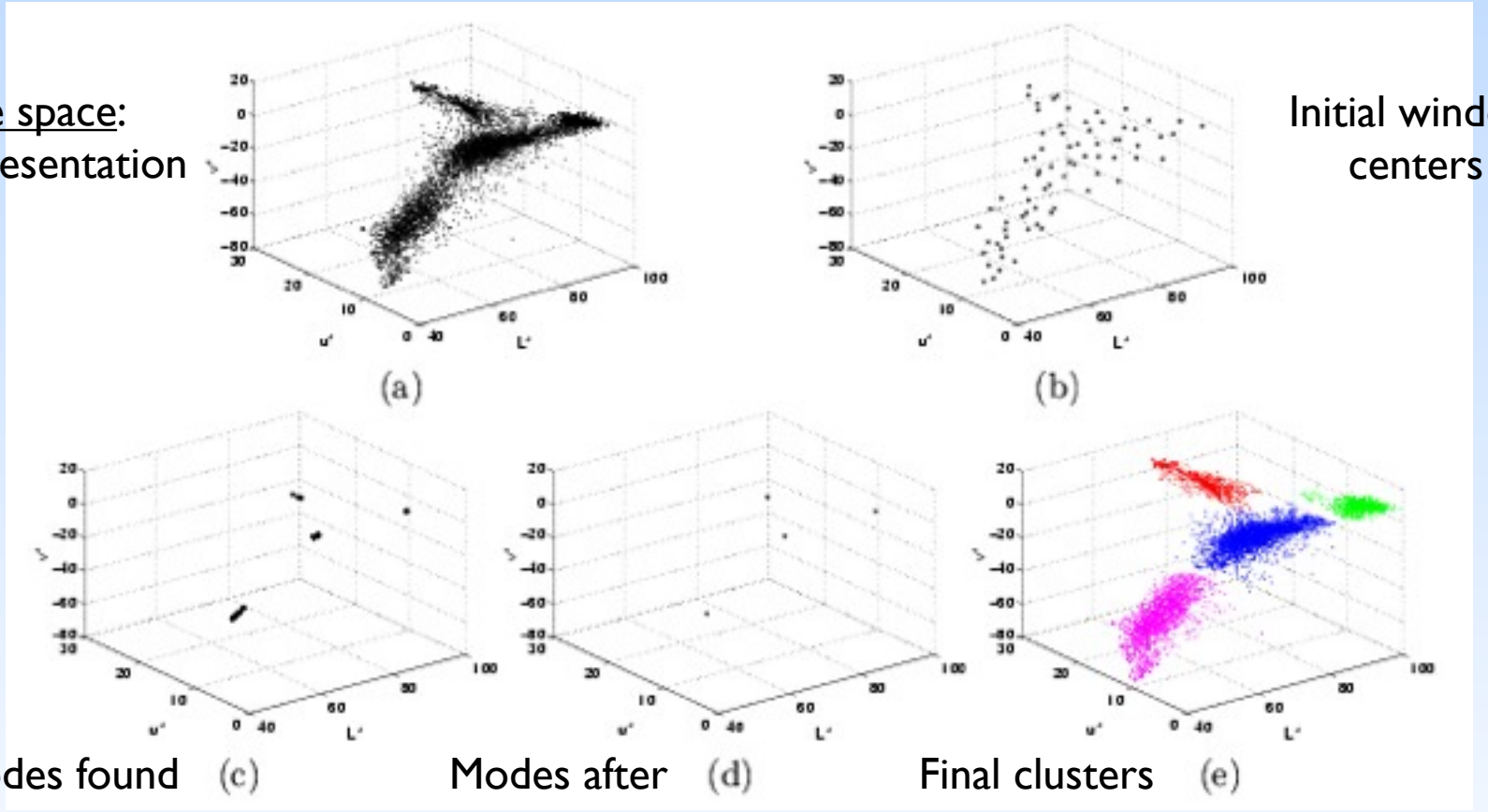
Feature space:
L*u*v representation



Clustering

Real Example

Feature space:
L*u*v representation



Initial window centers

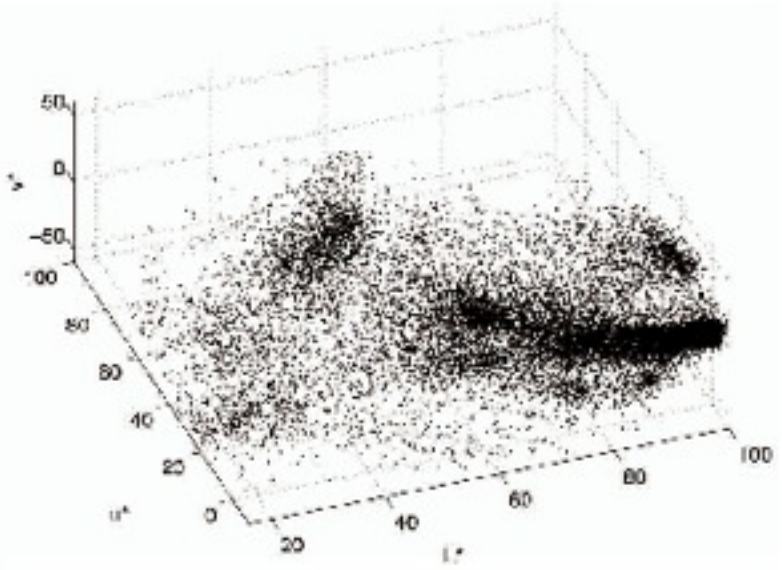
Modes found

Modes after pruning

Final clusters

Clustering

Real Example

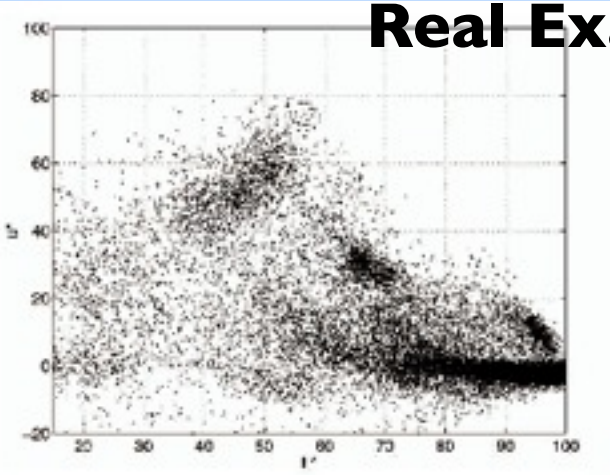


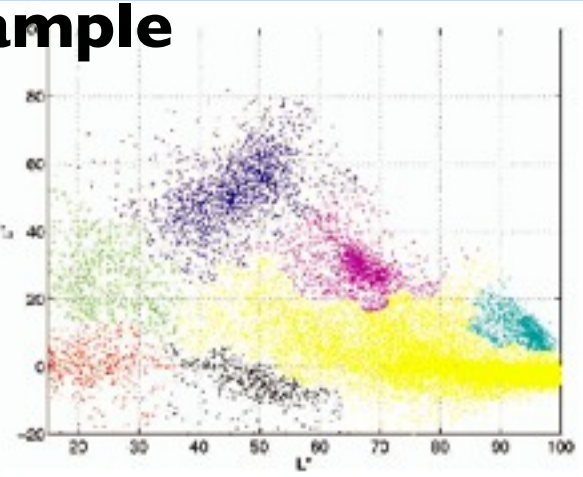
L*u*v space representation

Clustering

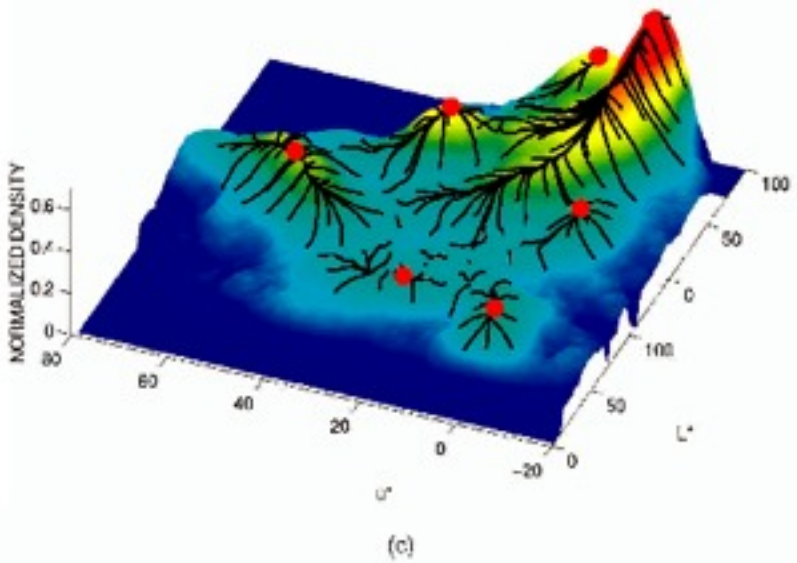
Real Example

2D (L*u) space representation





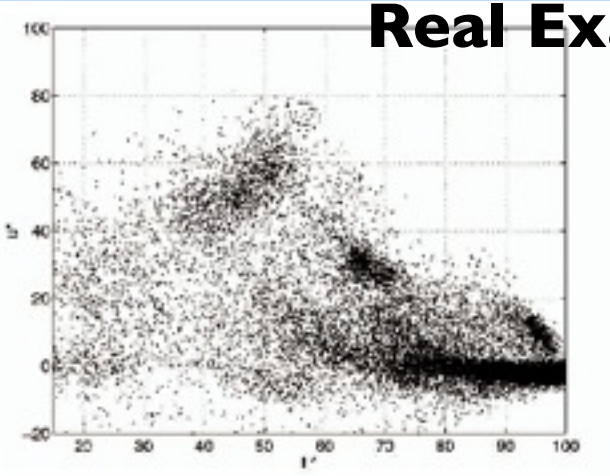
Final clusters

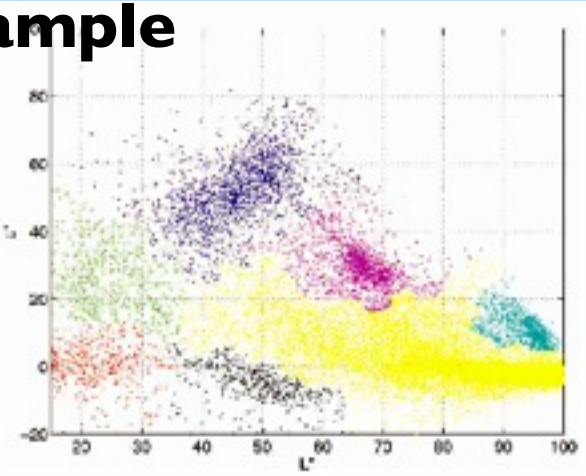


Clustering

Real Example

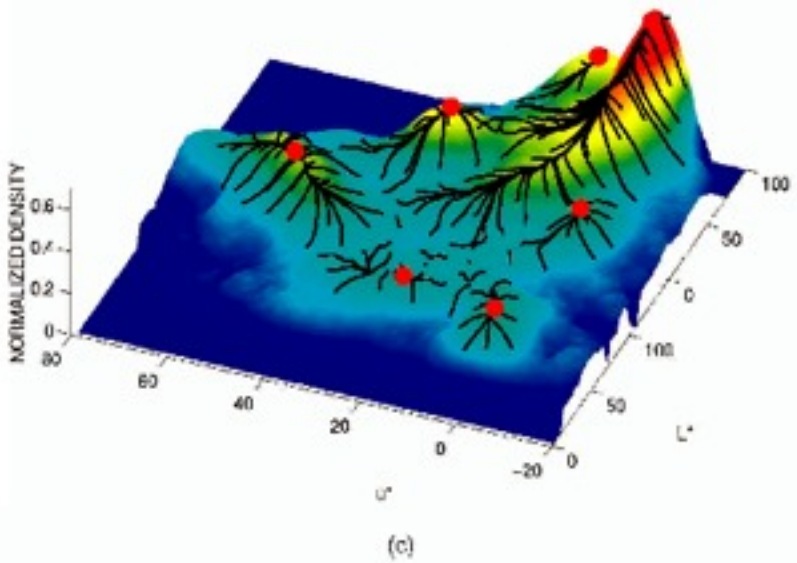
2D (L*u) space representation



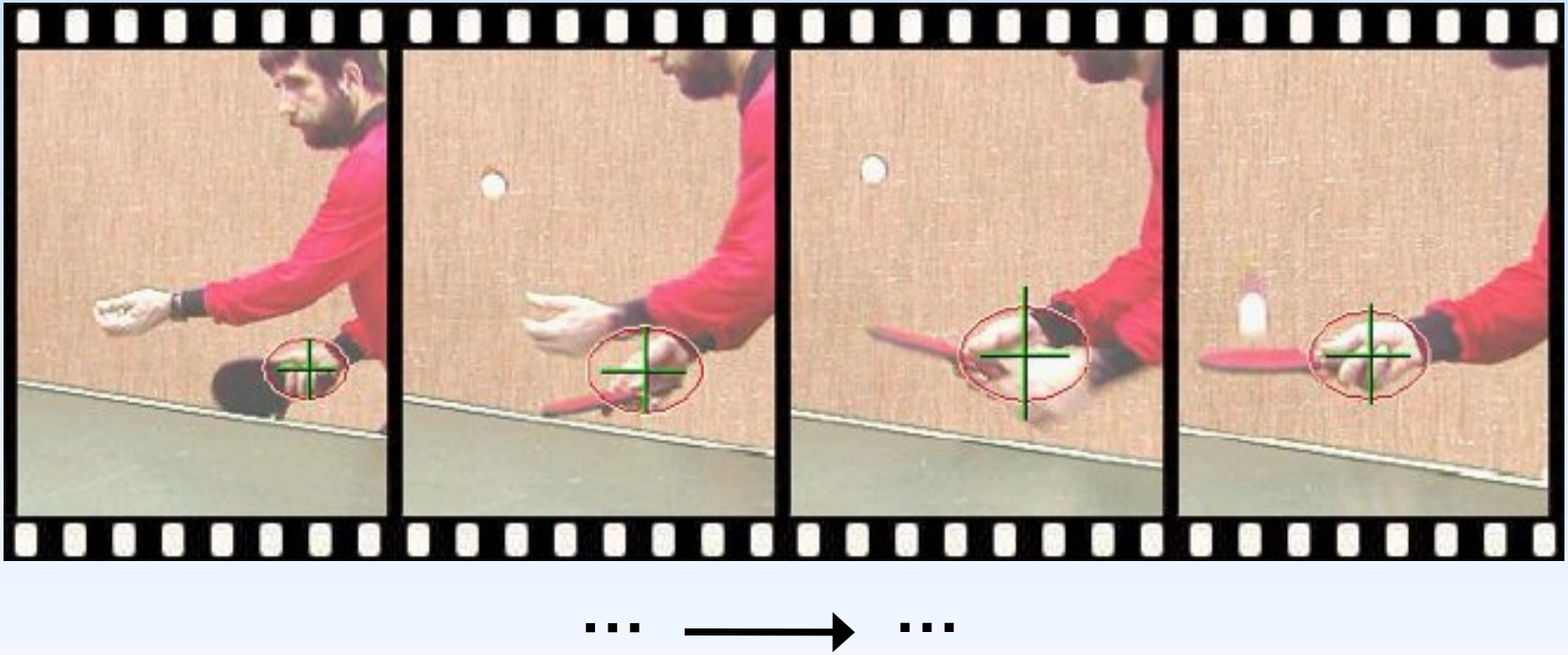


Final clusters

Not all trajectories in the attraction basin reach the same mode



Non-Rigid Object Tracking



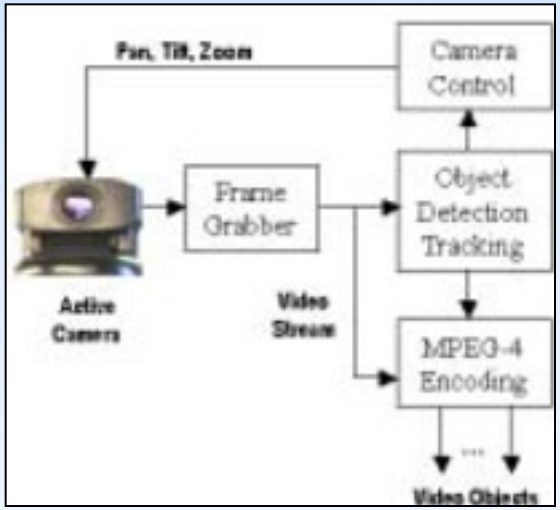
Non-Rigid Object Tracking

Real-Time

Surveillance

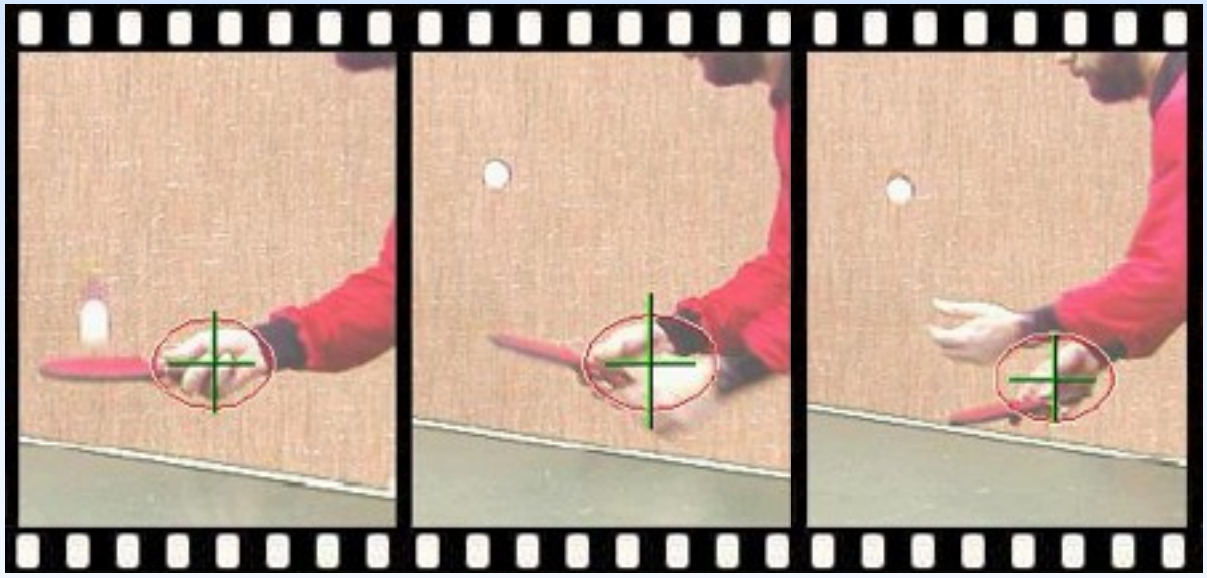
Driver Assistance

Object-Based Video Compression



Mean-Shift Object Tracking

General Framework: Target Representation

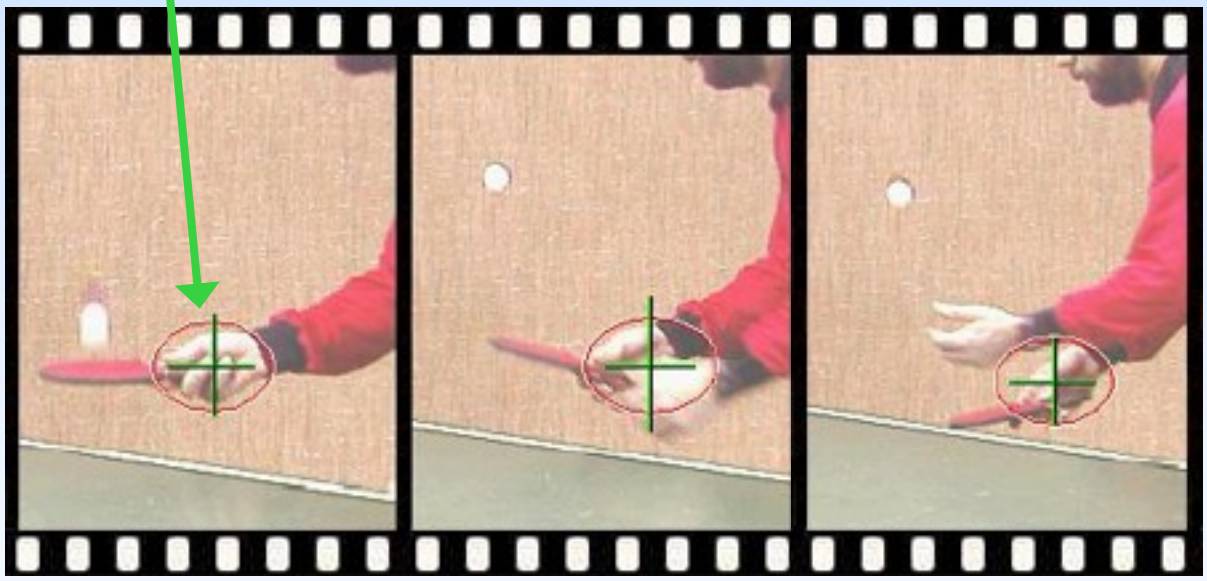


… Current frame → …

Mean-Shift Object Tracking

General Framework: Target Representation

Choose a reference model in the current frame



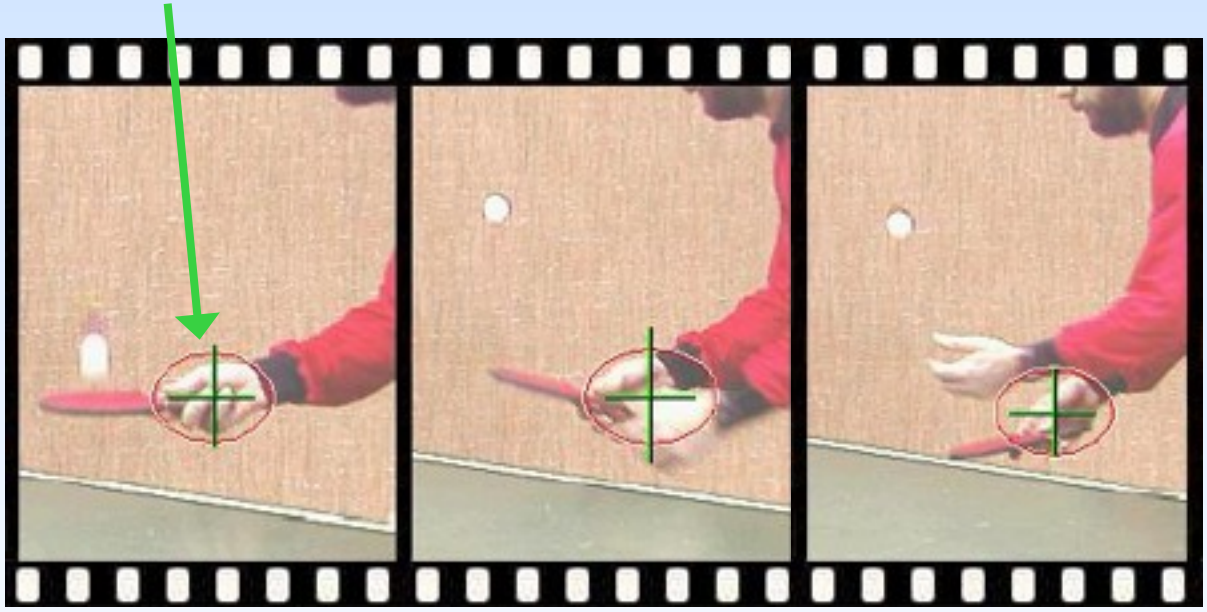
... Current frame → ...

Mean-Shift Object Tracking

General Framework: Target Representation

Choose a reference model in the current frame

Choose a feature space

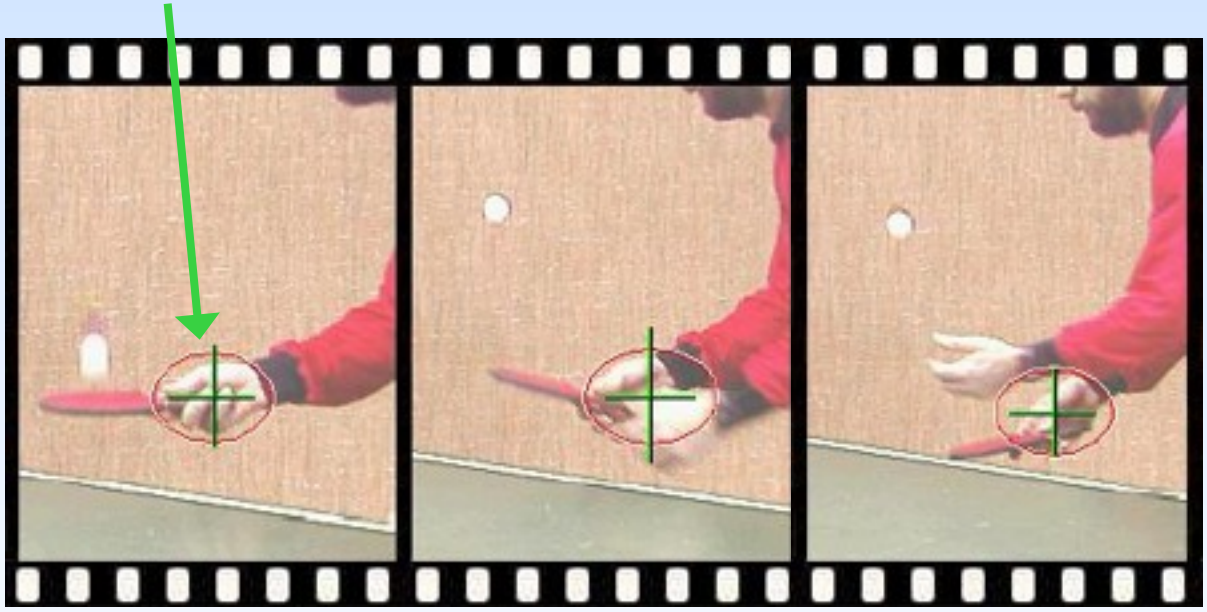


... Current frame ...

Mean-Shift Object Tracking

General Framework: Target Representation

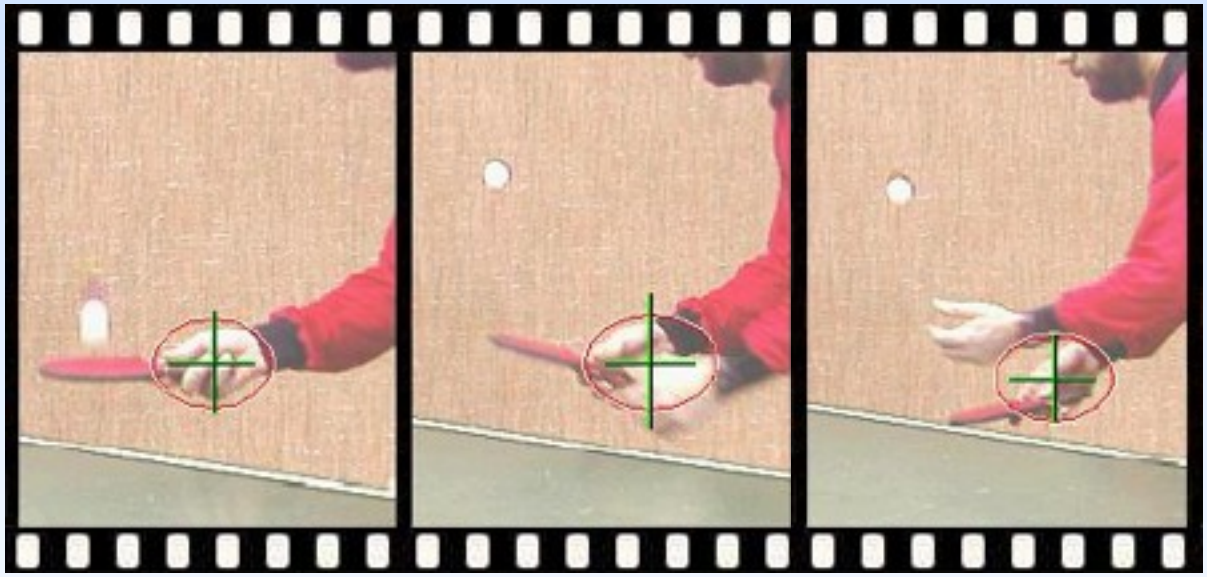
Choose a reference model in the current frame → Choose a feature space → Represent the model in the chosen feature space



… Current frame → …

Mean-Shift Object Tracking

General Framework: Target Representation



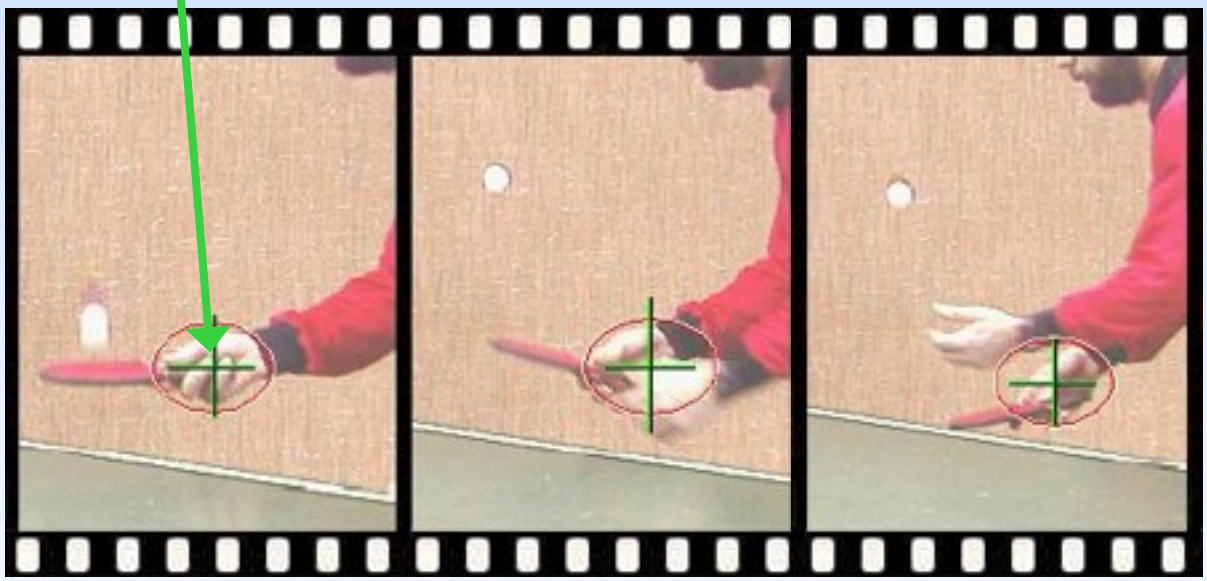
Model Candidate

… Current frame …

Mean-Shift Object Tracking

General Framework: Target Representation

Start from the position of the model in the current frame



Model Candidate

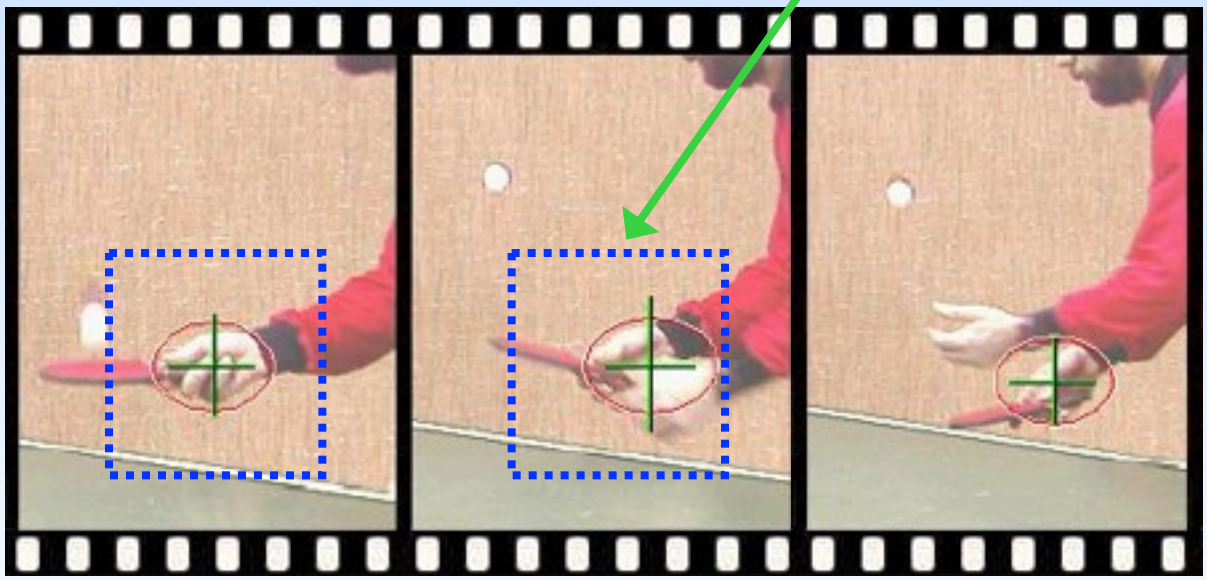
... Current frame ...

Mean-Shift Object Tracking

General Framework: Target Representation

Start from the position of the model in the current frame

Search in the model's neighborhood in next frame



Model

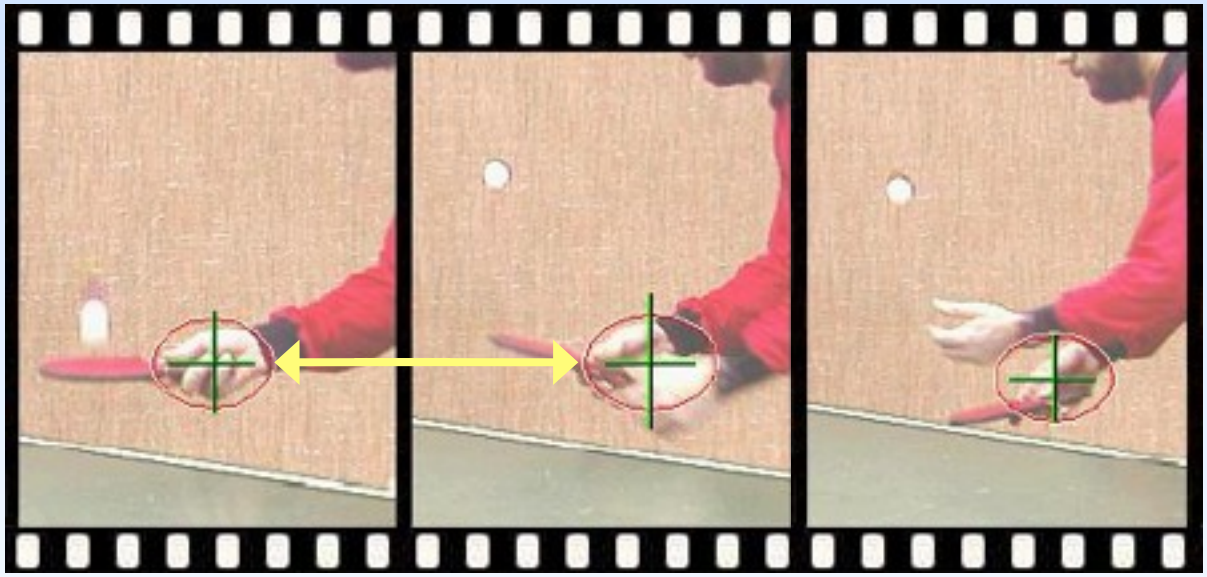
Candidate

... Current frame ...

Mean-Shift Object Tracking

General Framework: Target Representation

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.



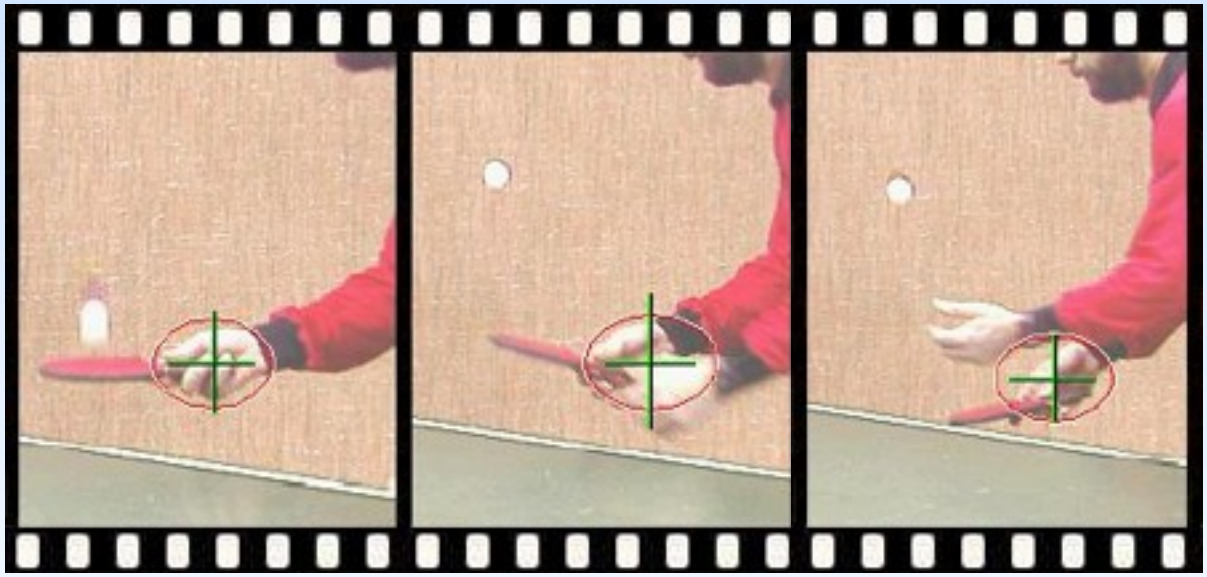
Model Candidate

... Current frame → ...

Mean-Shift Object Tracking

General Framework: Target Representation

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

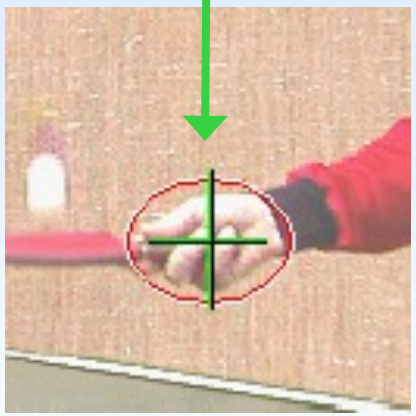
… Current frame → …

Repeat the same process in the next pair of frames

Mean-Shift Object Tracking

General Framework: Target Representation

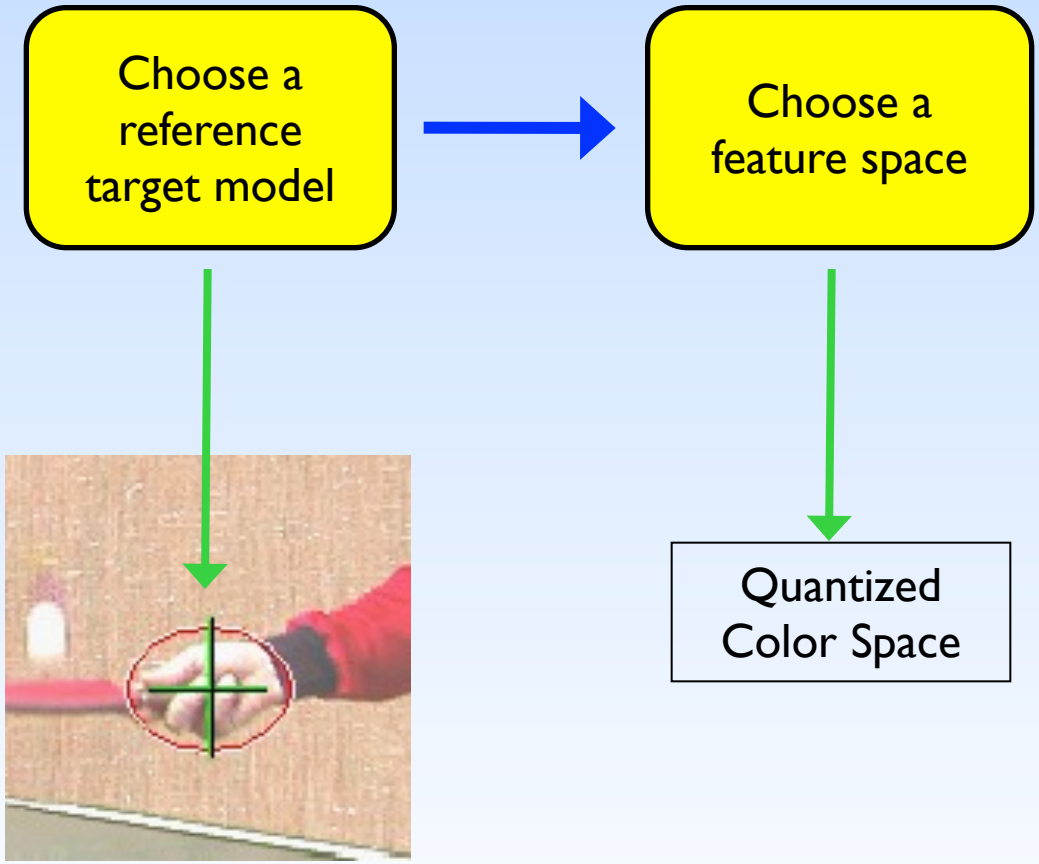
Choose a reference target model



Kernel Based Object Tracking, by Comaniniu, Ramesh, Meer

Mean-Shift Object Tracking

General Framework: Target Representation



Kernel Based Object Tracking, by Comaniniu, Ramesh, Meer

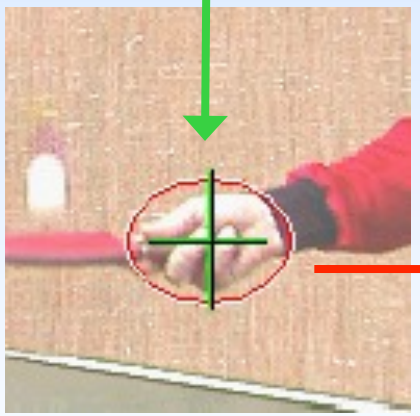
Mean-Shift Object Tracking

General Framework: Target Representation

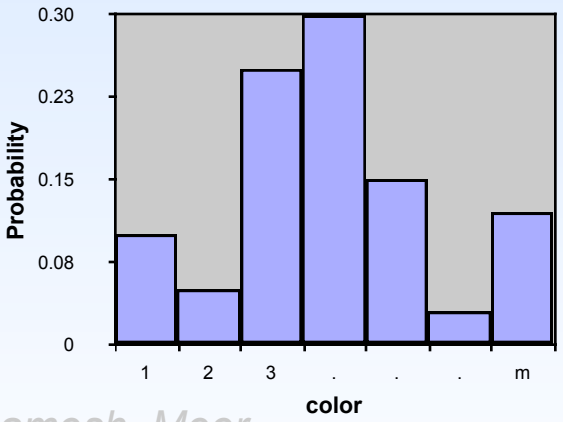
Choose a reference target model

Choose a feature space

Represent the model by its PDF in the feature space



Quantized Color Space

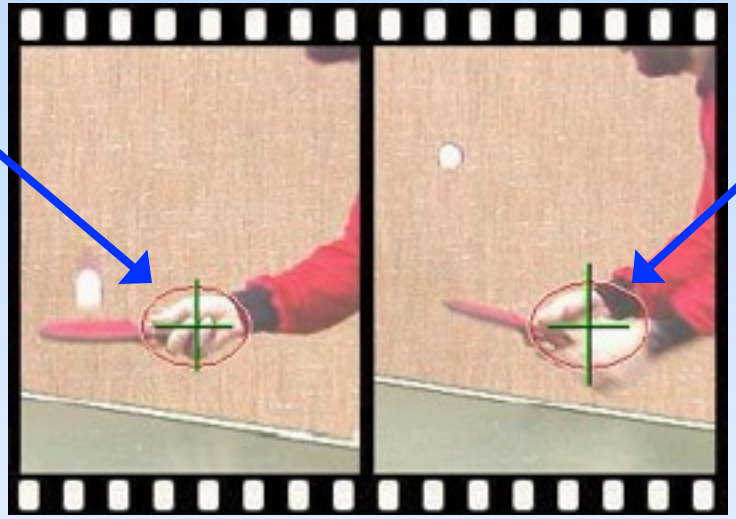


Kernel Based Object Tracking, by Comaniniu, Ramesh, Meer

Mean-Shift Object Tracking

PDF Representation

Target Model
(centered at 0)

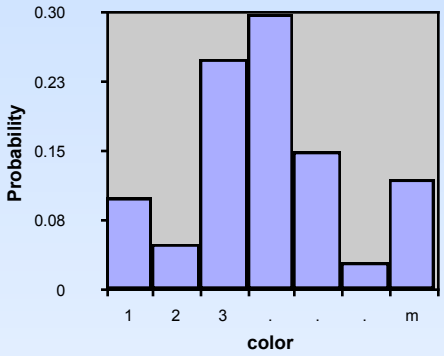


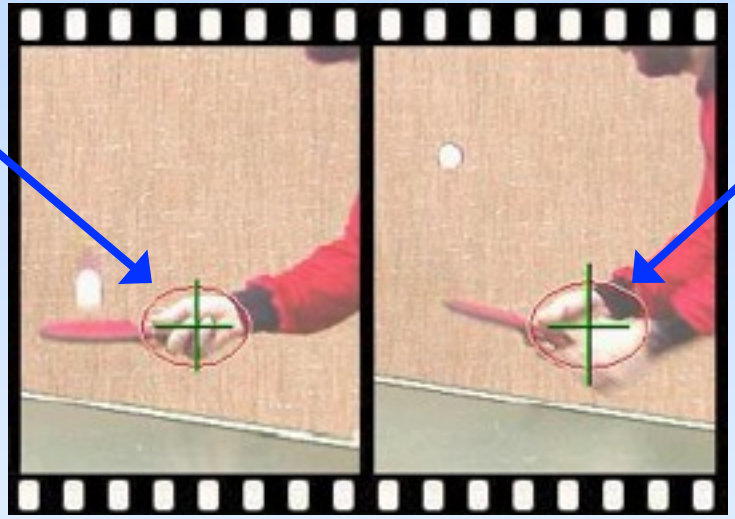
Target Candidate
(centered at y)

Mean-Shift Object Tracking

PDF Representation

Target Model
(centered at 0)





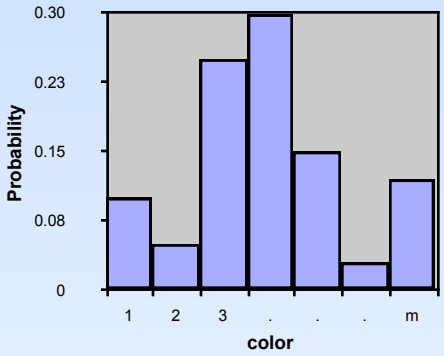
Target Candidate
(centered at y)

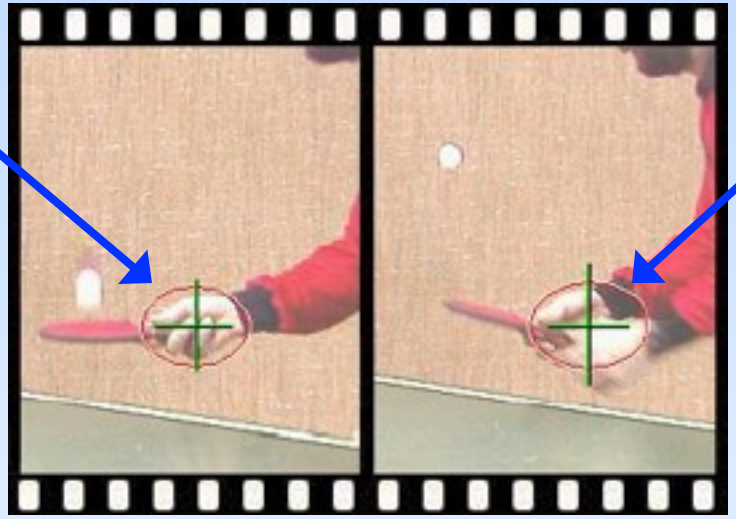
$$\vec{q} = \{q_u\}_{u=1..m} \qquad \sum_{u=1}^{m} q_u = 1$$

Mean-Shift Object Tracking

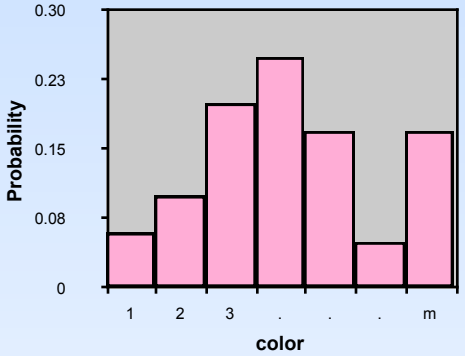
PDF Representation

Target Model
(centered at 0)





Target Candidate
(centered at y)



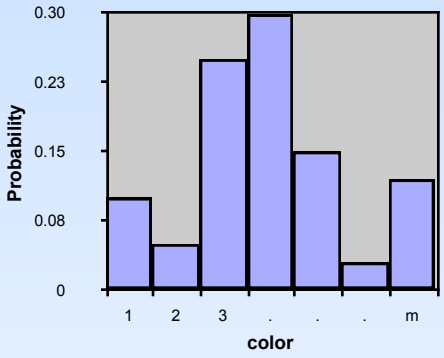
$$\vec{q} = \{q_u\}_{u=1..m} \qquad \sum_{u=1}^{m} q_u = 1$$

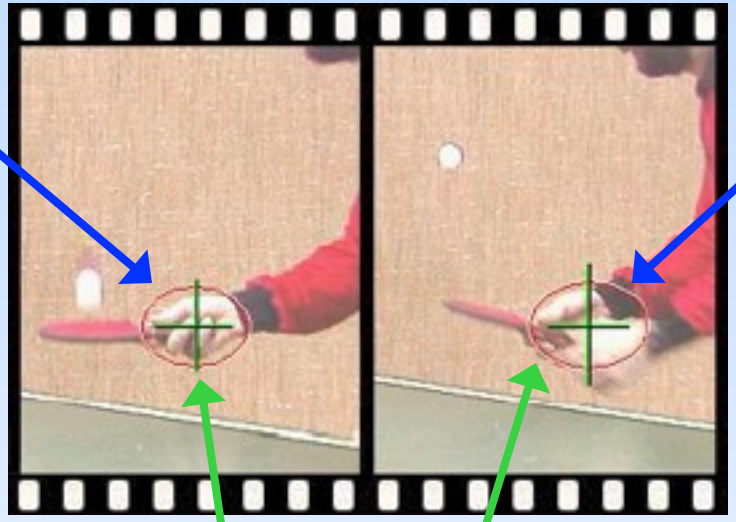
$$\vec{p}(y) = \{p_u(y)\}_{u=1..m} \qquad \sum_{u=1}^{m} p_u = 1$$

Mean-Shift Object Tracking

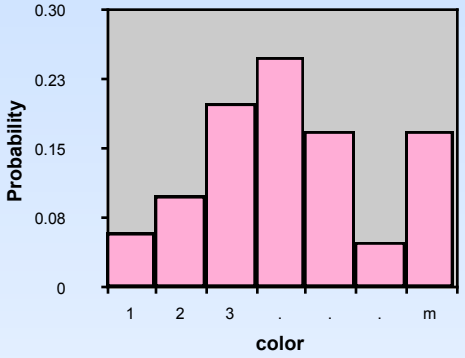
PDF Representation

Target Model
(centered at 0)





Target Candidate
(centered at y)



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

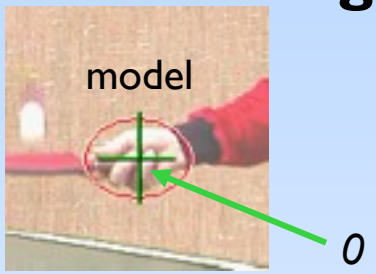
$$\vec{p}(y) = \{p_u(y)\}_{u=1..m} \qquad \sum_{u=1}^{m} p_u = 1$$

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

Mean-Shift Object Tracking

Finding the PDF of the target model

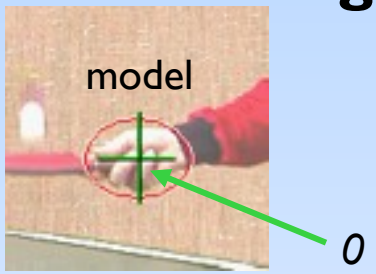
$\{x_i\}_{i=1..n}$ Target pixel locations



Mean-Shift Object Tracking

Finding the PDF of the target model

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



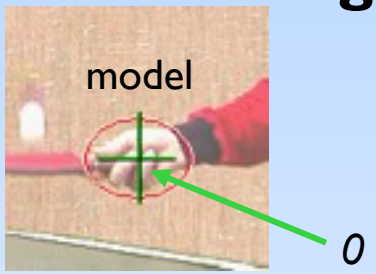
$k(x)$ A differentiable, isotropic, convex, monotonically decreasing kernel

- Peripheral pixels are affected by occlusion and background interference

Mean-Shift Object Tracking

Finding the PDF of the target model

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



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

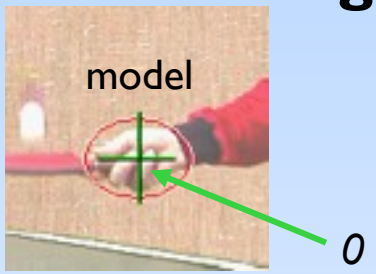
- Peripheral pixels are affected by occlusion and background interference

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

Mean-Shift Object Tracking

Finding the PDF of the target model

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

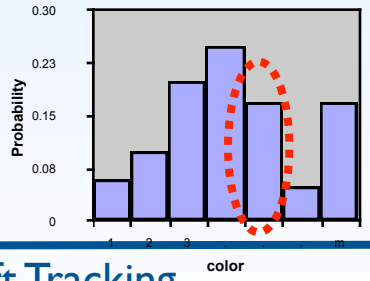


$k(x)$ A differentiable, isotropic, convex, monotonically decreasing kernel

- Peripheral pixels are affected by occlusion and background interference

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

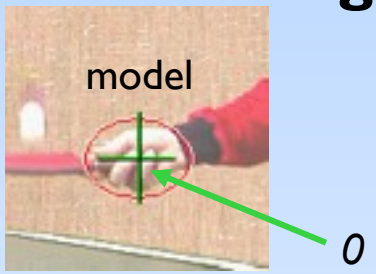
Probability of feature u in model



Mean-Shift Object Tracking

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

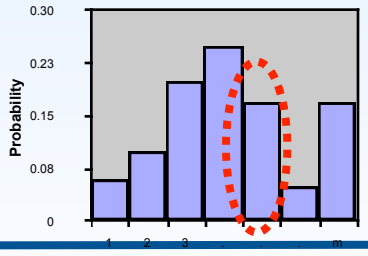
$b(x)$ The color bin index (1..*m*) of pixel *x*

Probability of feature u in model

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

Normalization factor

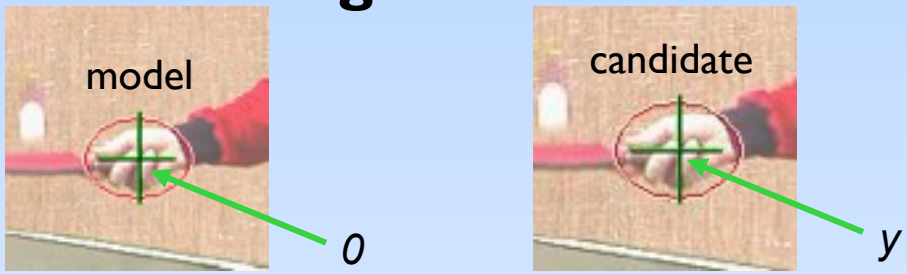
Pixel weight



Mean-Shift Object Tracking

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

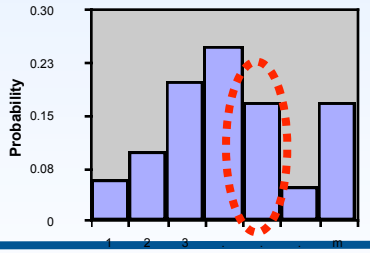
$b(x)$ The color bin index (1..*m*) of pixel *x*

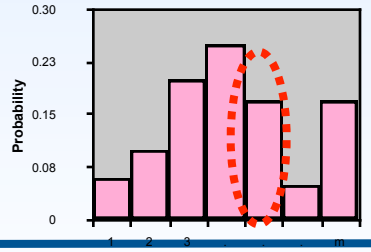
Probability of feature u in model **Probability of feature u in candidate**

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

Normalization factor

Pixel weight

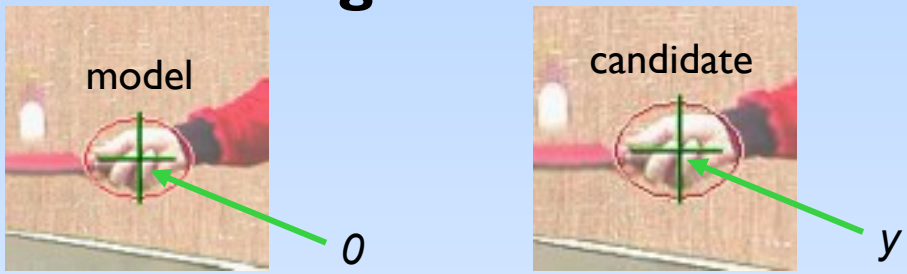




Mean-Shift Object Tracking

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

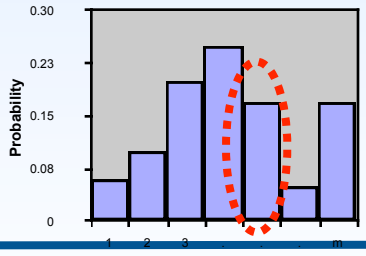
$b(x)$ The color bin index (1..*m*) of pixel *x*

Probability of feature u in model

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

Normalization factor

Pixel weight

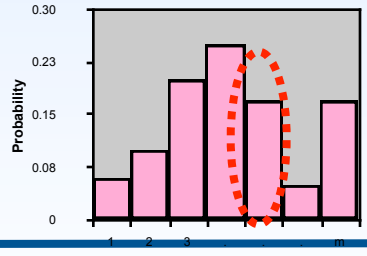


Probability of feature u in candidate

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

Normalization factor

Pixel weight



Mean-Shift Object Tracking

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

Mean-Shift Object Tracking

Similarity Function

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

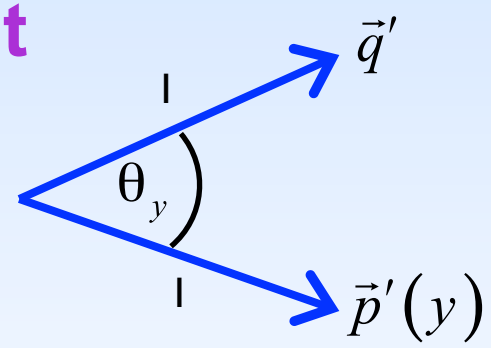
Target candidate: $\vec{p}(y) = (p_1(y), \ldots, p_m(y))$

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

The Bhattacharyya Coefficient

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

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



Mean-Shift Object Tracking

Similarity Function

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

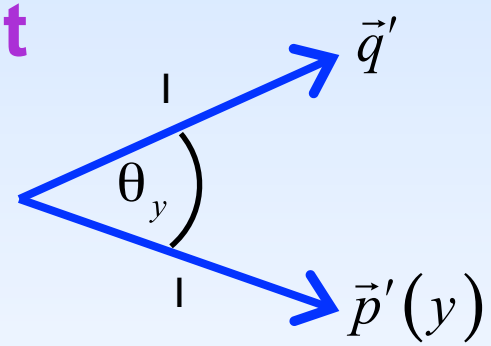
Target candidate: $\vec{p}(y) = (p_1(y), \ldots, p_m(y))$

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

The Bhattacharyya Coefficient

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

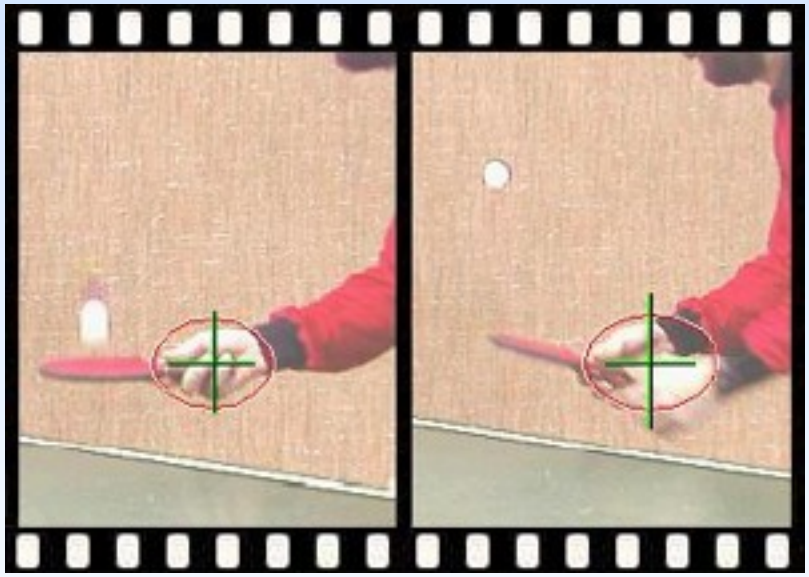
$$\vec{p}'(y) = \left(\sqrt{p_1(y)}, \ldots, \sqrt{p_m(y)}\right)$$



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

Mean-Shift Object Tracking

Target Localization Algorithm

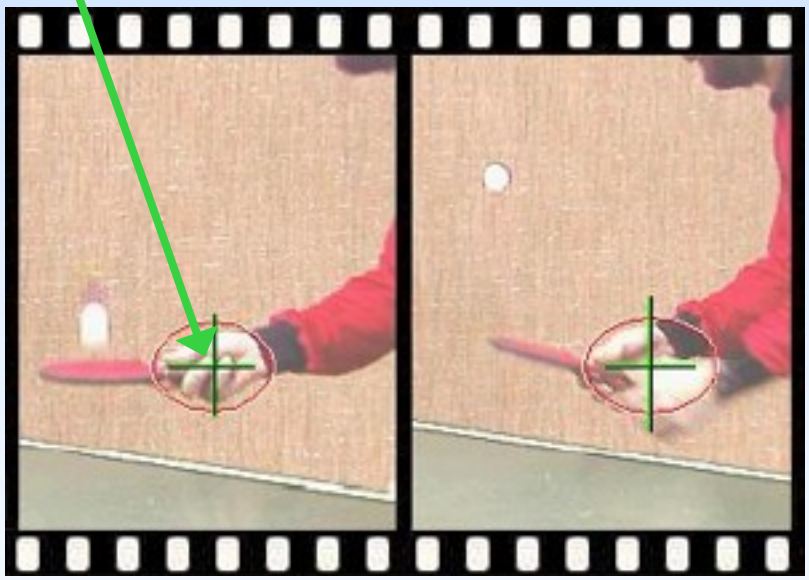


$\vec{q}$ $\vec{p}(y)$

Mean-Shift Object Tracking

Target Localization Algorithm

Start from the position of the model in the current frame



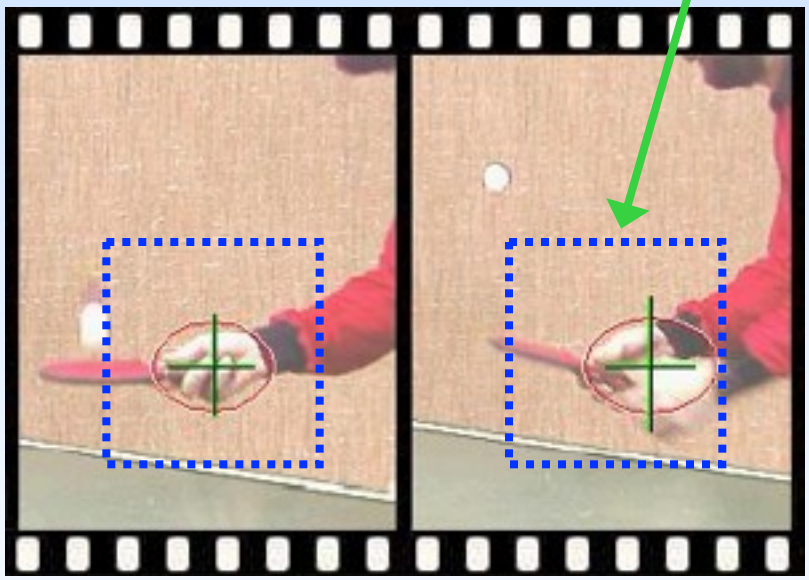
$\vec{q}$ $\vec{p}(y)$

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



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

Mean-Shift Object Tracking

Target Localization Algorithm

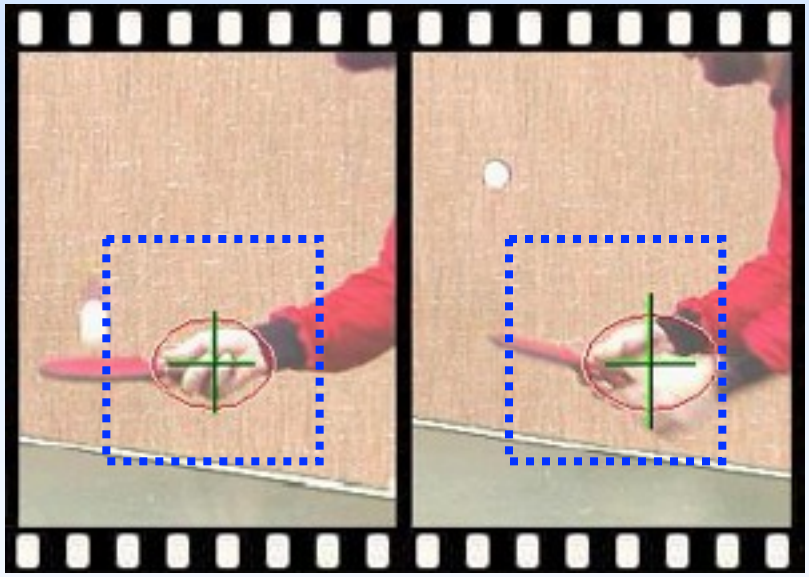
Start from the position of the model in the current frame

→

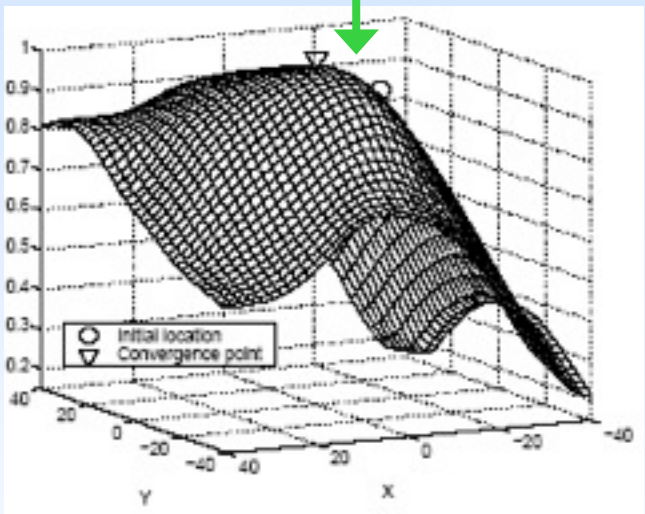
Search in the model's neighborhood in next frame

→

Find best candidate by maximizing a similarity func.



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



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

Mean-Shift Object Tracking

Approximating the Similarity Function

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

Model location: y_0

Candidate location: y

Mean-Shift Object Tracking

Approximating the Similarity Function

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

Model location: y_0

Candidate location: y

Linear approx. (around y_0)

$$f(y) \approx \frac{1}{2} \sum_{u=1}^{m} \sqrt{p_u(y_0) q_u} + \frac{1}{2} \sum_{u=1}^{m} p_u(y) \sqrt{\frac{q_u}{p_u(y_0)}}$$

Mean-Shift Object Tracking

Approximating the Similarity Function

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

Model location: y_0

Candidate location: y

Linear approx. (around y_0)

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

Mean-Shift Object Tracking

Approximating the Similarity Function

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

Model location: y_0

Candidate location: y

Linear approx. (around y_0)

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

Mean-Shift Object Tracking

Approximating the Similarity Function

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

Model location: y_0

Candidate location: y

Linear approx. (around y_0)

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

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

Mean-Shift Object Tracking

Approximating the Similarity Function

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

Model location: y_0

Candidate location: y

Linear approx. (around y_0)

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

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

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

Mean-Shift Object Tracking

Approximating the Similarity Function

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

Model location: y_0

Candidate location: y

Linear approx. (around y_0)

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

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

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

Mean-Shift Object Tracking

Approximating the Similarity Function

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

Model location: y_0

Candidate location: y

Linear approx. (around y_0)

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

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

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

Mean-Shift Object Tracking

Approximating the Similarity Function

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

Model location: y_0

Candidate location: y

Linear approx. (around y_0)

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

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

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

Density estimate! (as a function of y)

Mean-Shift Object Tracking

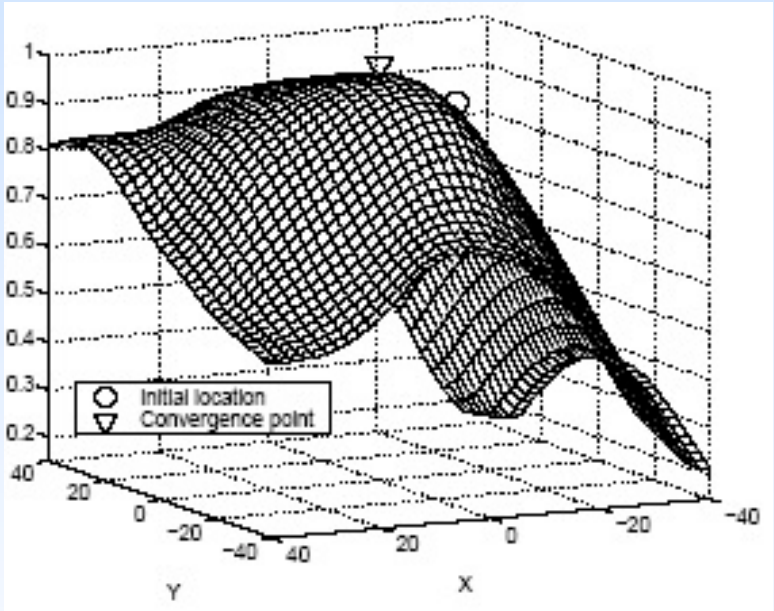
Maximizing the Similarity Function

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

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

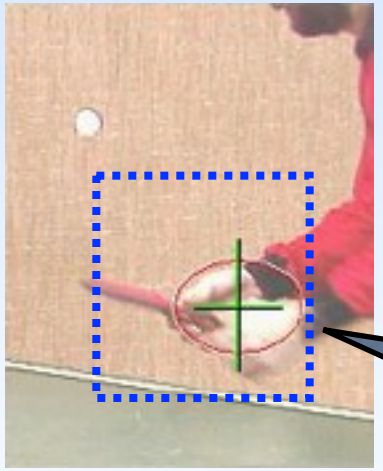


Mean-Shift Object Tracking

Maximizing the Similarity Function

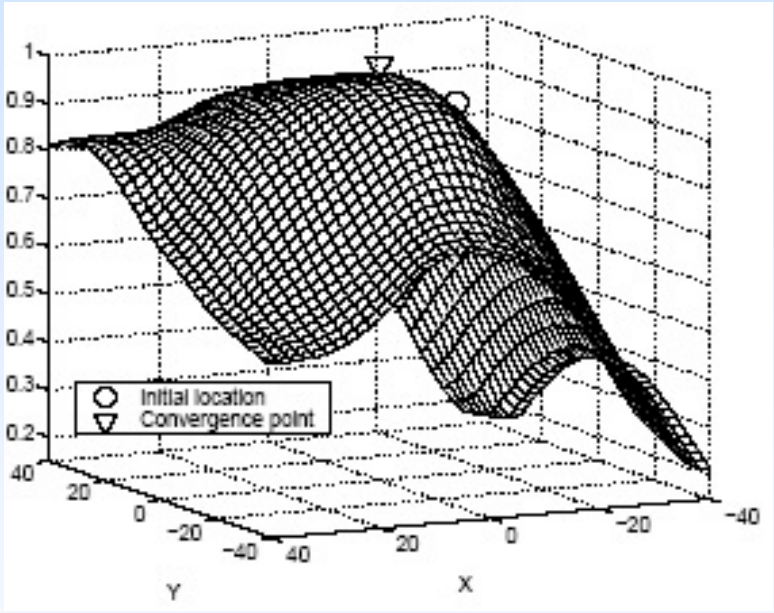
The mode of $\frac{C_h}{2}\sum_{i=1}^{n} w_i k\left(\left\|\frac{y-x_i}{h}\right\|^2\right)$ = sought maximum

Important Assumption:



The target representation provides sufficient discrimination

One mode in the searched neighborhood



Mean-Shift Object Tracking

Applying Mean-Shift

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

Mean-Shift Object Tracking

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

Mean-Shift Object Tracking

Applying Mean-Shift

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

About Kernels and Profiles

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

About Kernels and Profiles

A special class of radially symmetric kernels:

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

The profile of kernel *K*

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

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

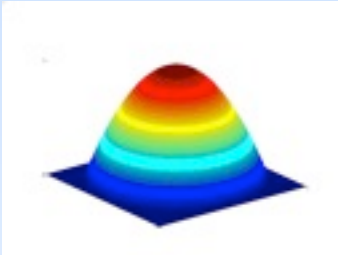
Mean-Shift Object Tracking

Choosing the Kernel

A special class of radially symmetric kernels:

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

Epanechnikov kernel:



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

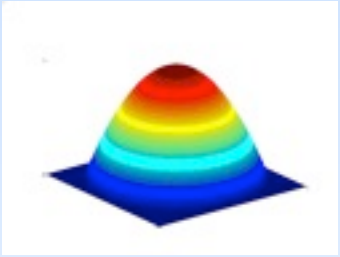
Mean-Shift Object Tracking

Choosing the Kernel

A special class of radially symmetric kernels:

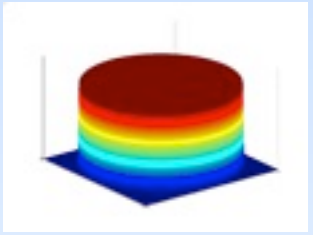
$$K(x) = ck\left(\|x\|^2\right)$$

Epanechnikov kernel:



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

Uniform kernel:



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

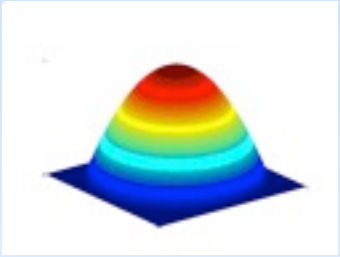
Mean-Shift Object Tracking

Choosing the Kernel

A special class of radially symmetric kernels:

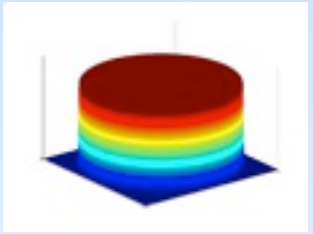
$$K(x) = ck\left(\|x\|^2\right)$$

Epanechnikov kernel:



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

Uniform kernel:



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

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

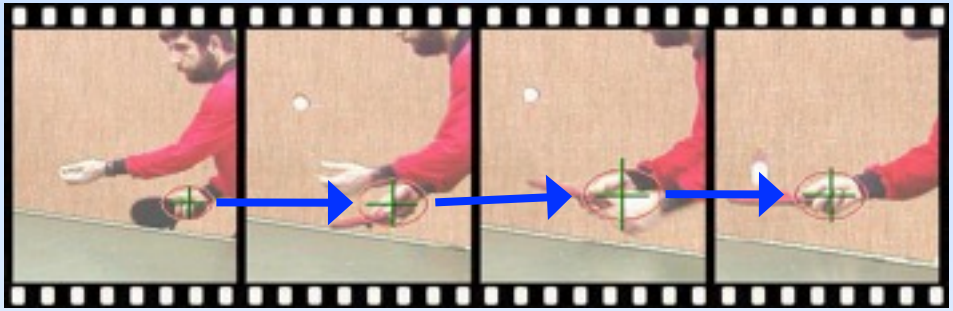
Mean-Shift Object Tracking

Adaptive Scale

Problem

:

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

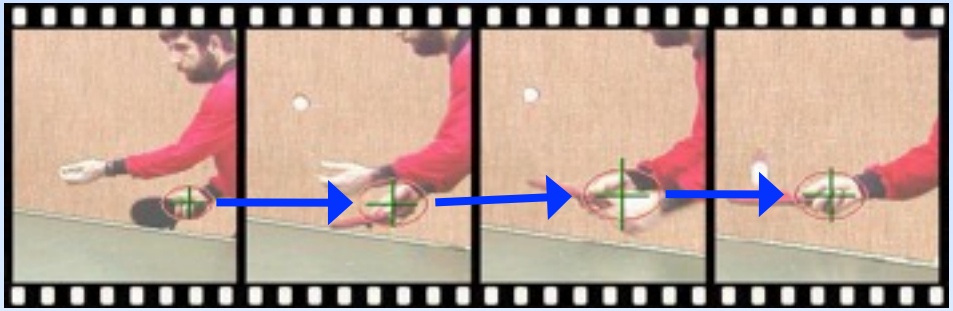


Mean-Shift Object Tracking

Adaptive Scale

Problem:

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



Solution:

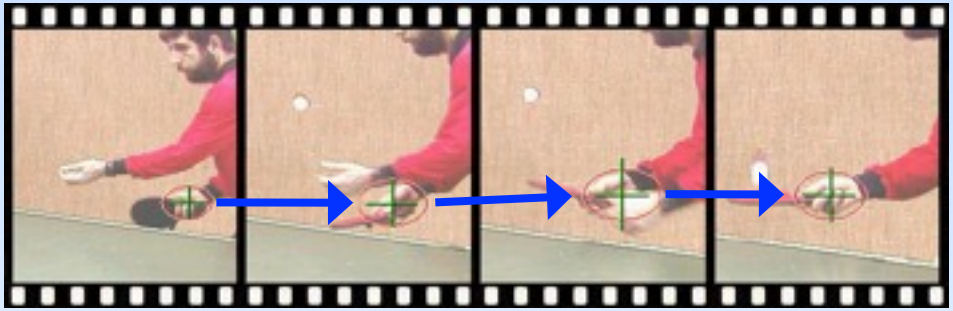
Run localization 3 times with different *h*

Mean-Shift Object Tracking

Adaptive Scale

Problem :

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



Solution :

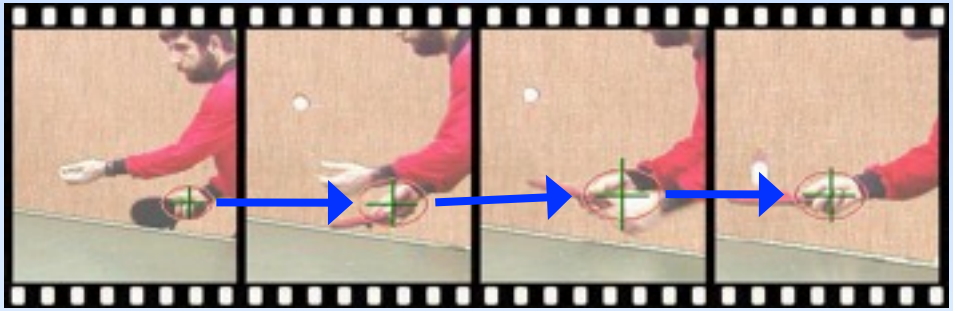
Run localization 3 times with different *h*

Mean-Shift Object Tracking

Adaptive Scale

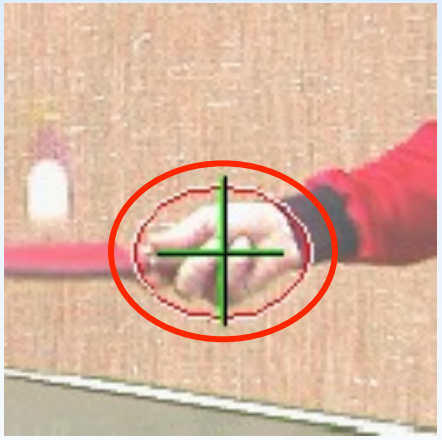
Problem :

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



Solution :

Run localization 3 times with different *h*

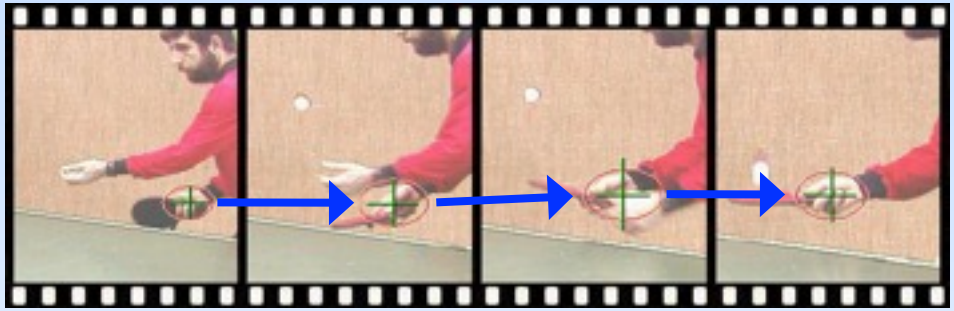


Mean-Shift Object Tracking

Adaptive Scale

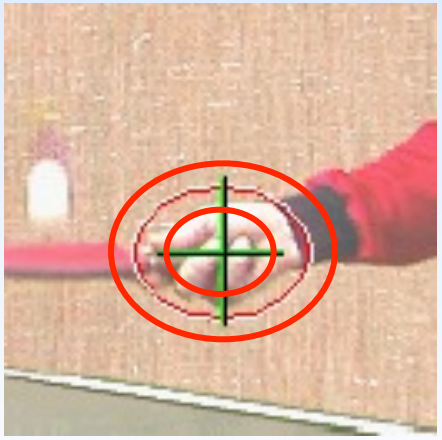
Problem :

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



Solution :

Run localization 3 times with different *h*

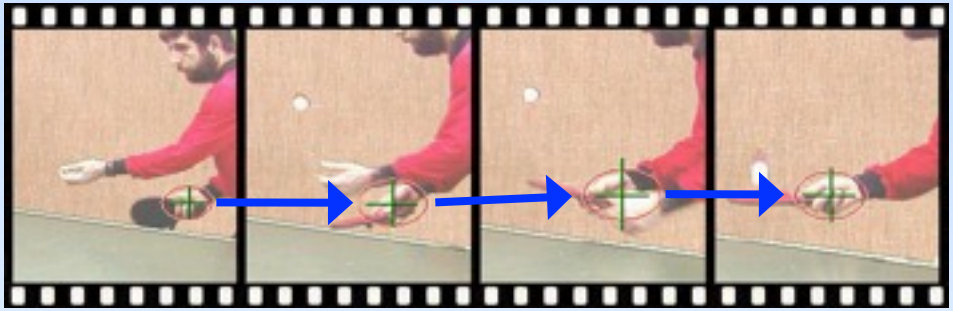


Mean-Shift Object Tracking

Adaptive Scale

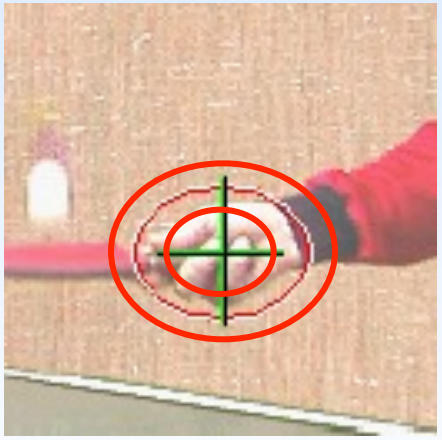
Problem

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



Solution

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



Mean-Shift Object Tracking

Results

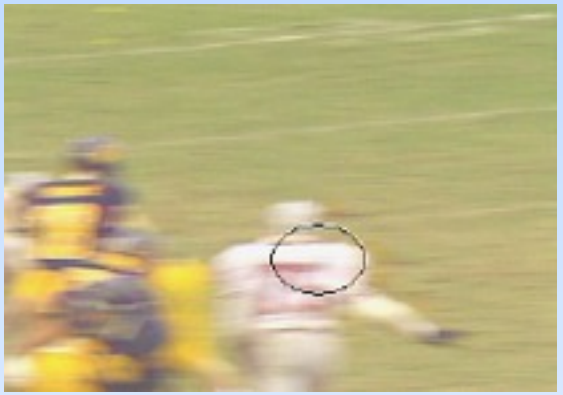
Feature space: 16×16×16 quantized RGB
Target: manually selected on 1st frame
Average mean-shift iterations: 4

Mean-Shift Object Tracking

Results

Partial occlusion

Distraction



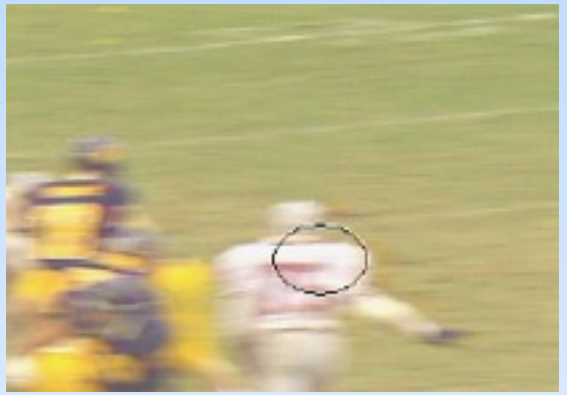
Motion blur

Mean-Shift Object Tracking

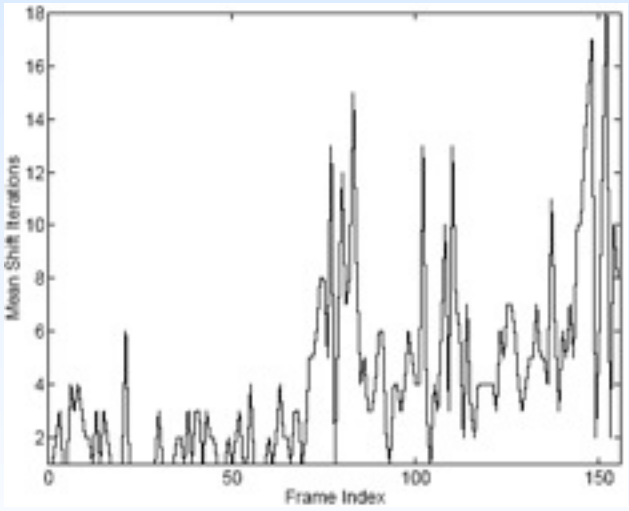
Results

Partial occlusion

Distraction



Motion blur



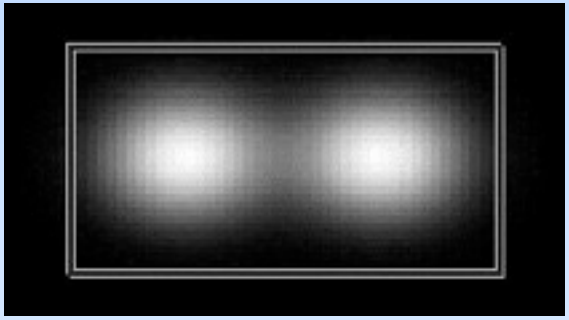
Mean-Shift Object Tracking

Results

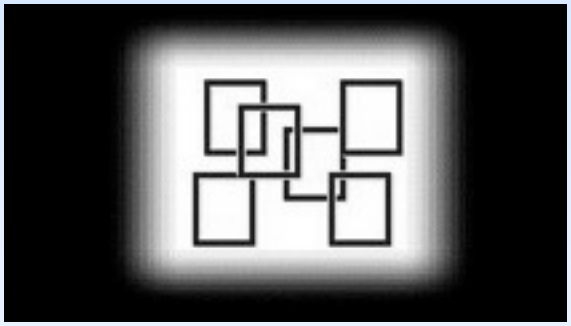
Mean-Shift Object Tracking

The Scale Selection Problem

Kernel too big



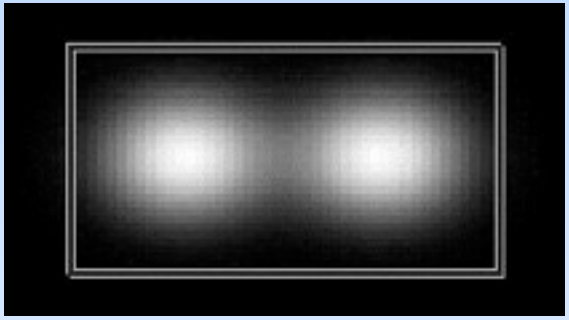
Kernel too small



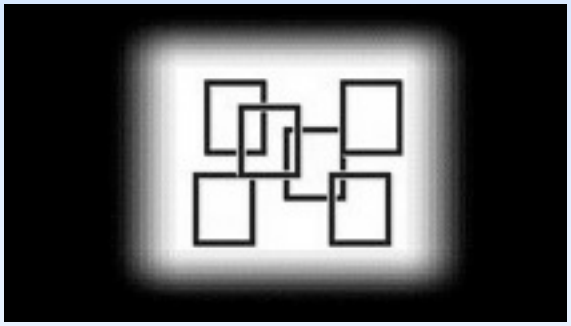
Mean-Shift Object Tracking

The Scale Selection Problem

Kernel too big



Kernel too small



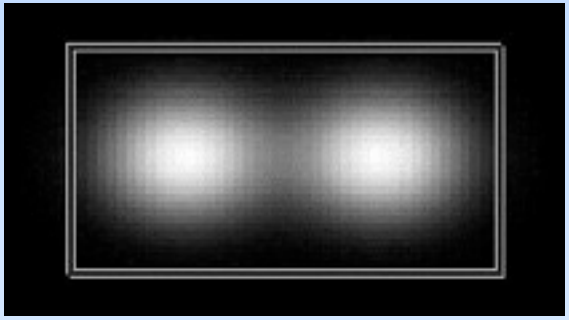
Poor localization

h mustn't get too big or too small!

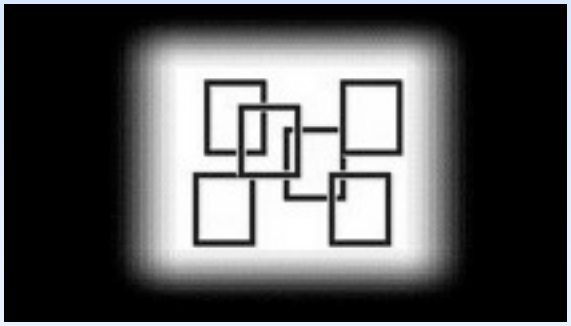
Mean-Shift Object Tracking

The Scale Selection Problem

Kernel too big



Kernel too small



Poor localization → *h* mustn't get too big or too small!

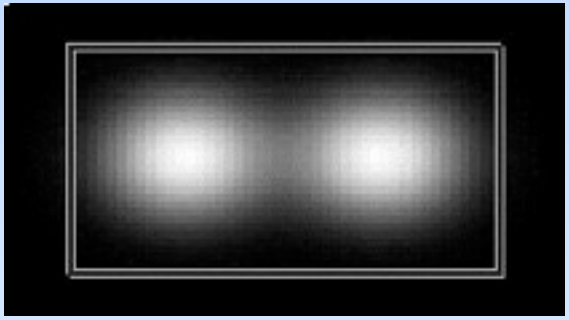
Problem:

In uniformly colored regions, similarity is invariant to *h*

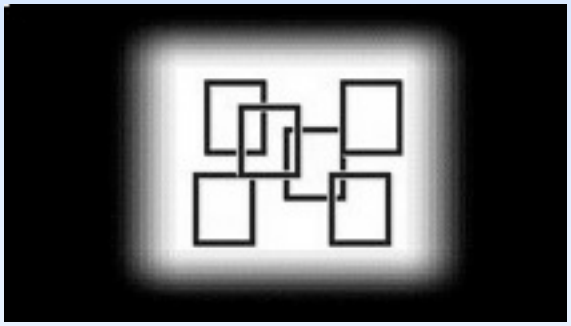
Mean-Shift Object Tracking

The Scale Selection Problem

Kernel too big



Kernel too small



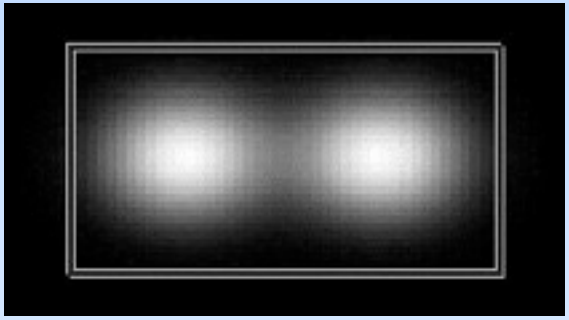
Poor localization → *h* mustn't get too big or too small!

Problem: In uniformly colored regions, similarity is invariant to *h* → Smaller *h* may achieve better similarity

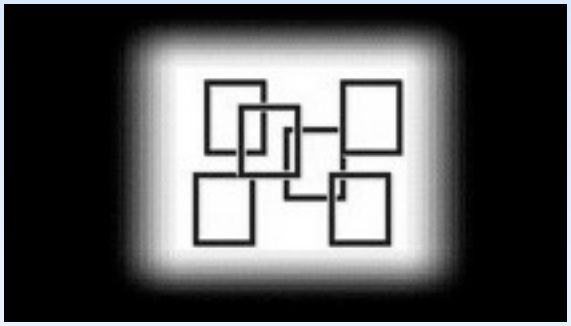
Mean-Shift Object Tracking

The Scale Selection Problem

Kernel too big



Kernel too small



Poor localization → *h* mustn't get too big or too small!

Problem:

In uniformly colored regions, similarity is invariant to *h* → Smaller *h* may achieve better similarity → Nothing keeps *h* from shrinking too small!

Tracking Through Scale Space

Motivation

Spatial localization for several scales

Previous method

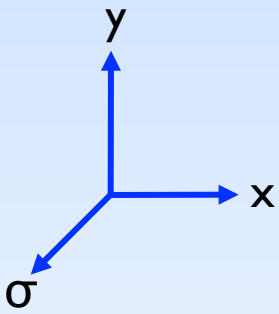
Simultaneous localization in space and scale

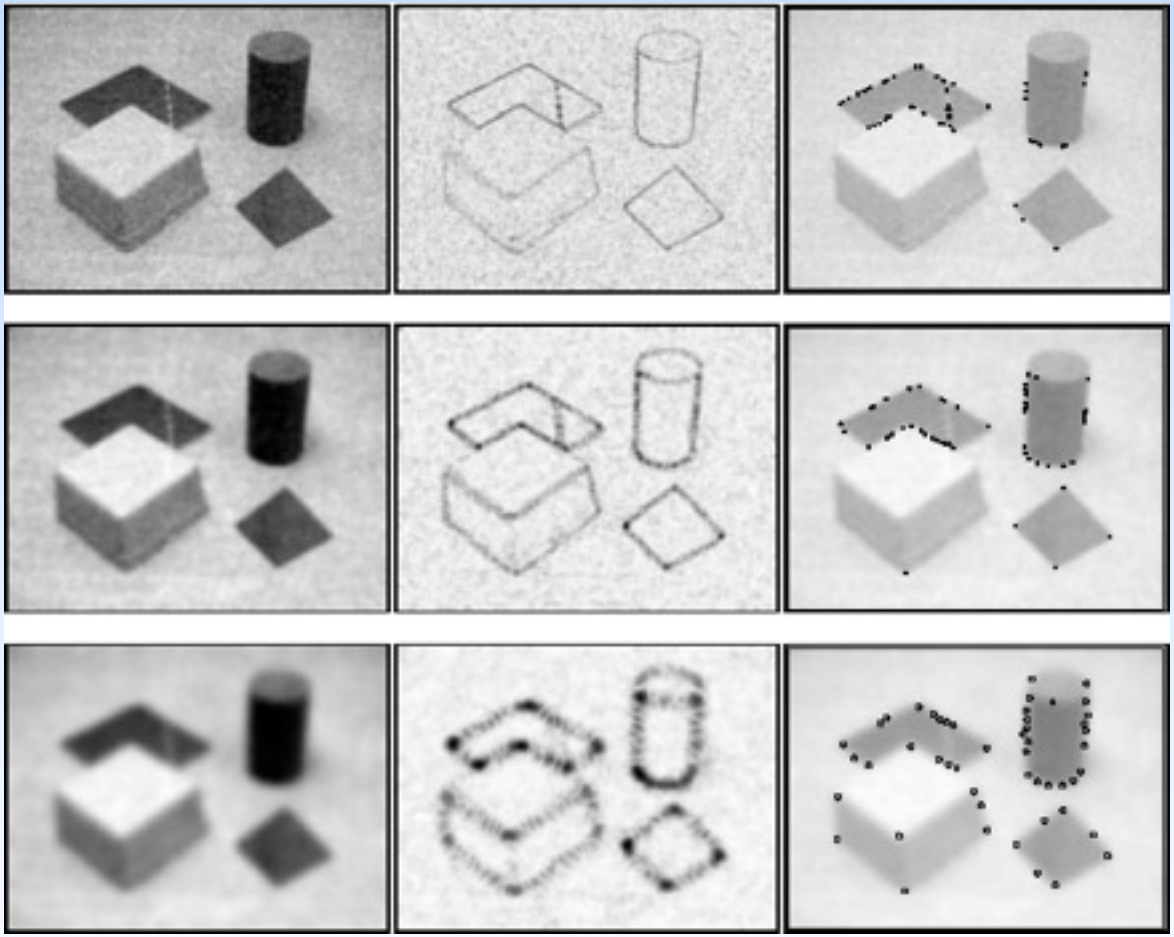
This method

Mean-shift Blob Tracking through Scale Space, by R. Collins

Lindeberg's Theory

Selecting the best scale for describing image features





Scale-space representation | Differential operator applied | 50 strongest responses

Lindeberg's Theory

Selecting the best scale for describing image features

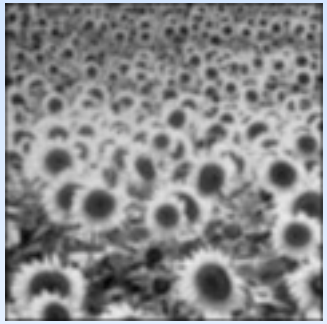
$f(x) =$ 

Lindeberg's Theory

Selecting the best scale for describing image features

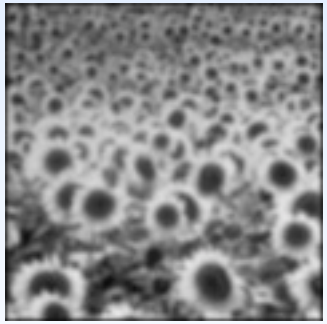
$G(x;\sigma_1)$

$G(x;\sigma_2)$



$\vdots$

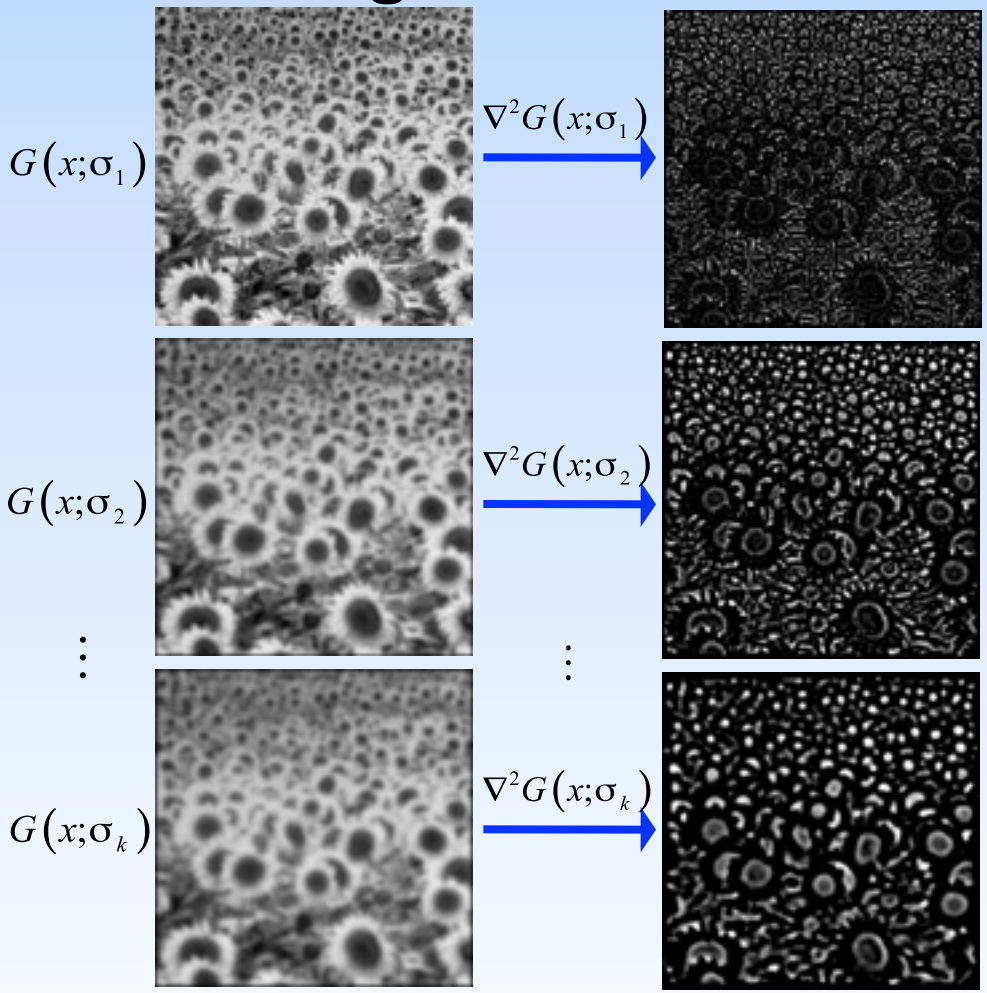
$G(x;\sigma_k)$



Scale-space representation

Lindeberg's Theory

Selecting the best scale for describing image features



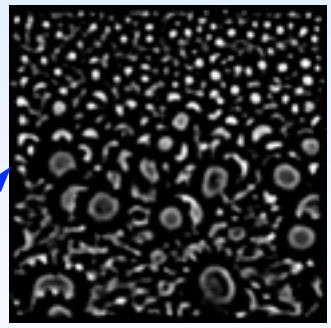
Lindeberg's Theory

Selecting the best scale for describing image features

$LOG(x;\sigma_1)$

$LOG(x;\sigma_2)$

$LOG(x;\sigma_k)$



Laplacian of Gaussian (LOG)

2D LOG filter with scale σ

$$LOG(x;\sigma) = \frac{2\sigma^2 - \|x\|^2}{2\pi\sigma^6} e^{\frac{-\|x\|^2}{2\sigma^2}}$$

Lindeberg's Theory

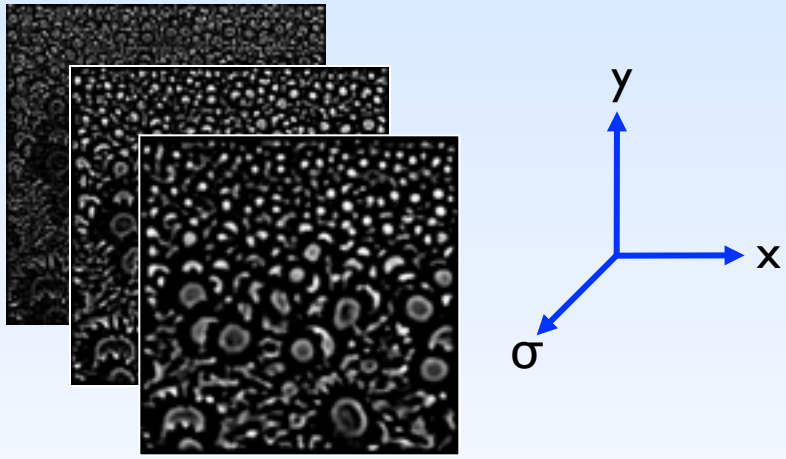
Selecting the best scale for describing image features

2D LOG filter with scale σ

$$LOG(x;\sigma) = \frac{2\sigma^2 - \|x\|^2}{2\pi\sigma^6} e^{\frac{-\|x\|^2}{2\sigma^2}}$$

3D scale-space representation

$$\forall x \in f,\ \forall \sigma_{1..k}:$$
$$L(x,\sigma) = LOG(x;\sigma) * f(x)$$



Lindeberg's Theory

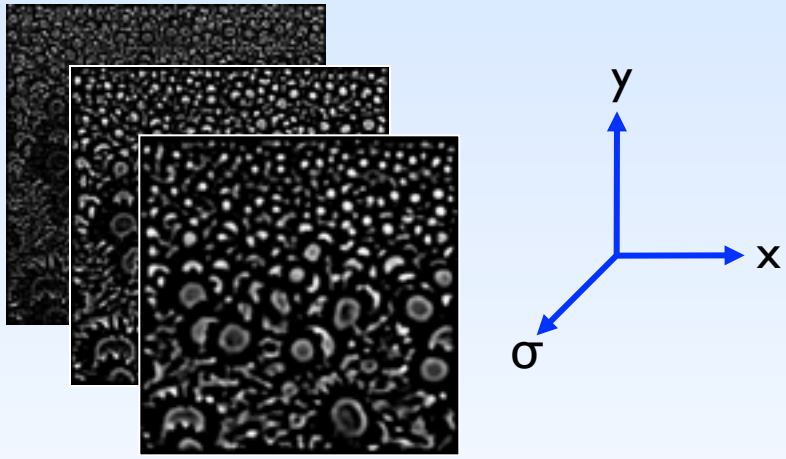
Selecting the best scale for describing image features

2D LOG filter with scale σ

$$LOG(x;\sigma) = \frac{2\sigma^2 - \|x\|^2}{2\pi\sigma^6} e^{\frac{-\|x\|^2}{2\sigma^2}}$$

3D scale-space representation

$$\forall x \in f,\ \forall \sigma_{1..k}:$$
$$L(x,\sigma) = LOG(x;\sigma) * f(x)$$



Best features are at (x,σ) that maximize L

Lindeberg's Theory

Multi-Scale Feature Selection Process

Original Image

$f(x)$

Lindeberg's Theory

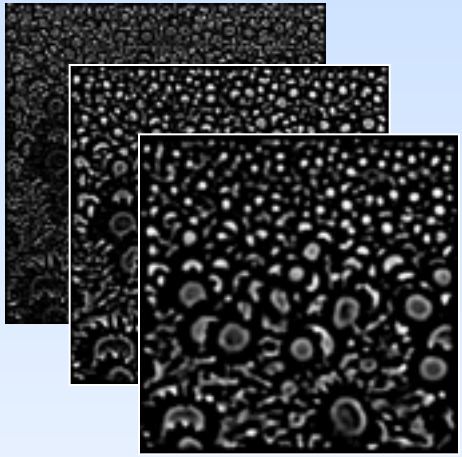
Multi-Scale Feature Selection Process

Original Image

$f(x)$

Convolve

3D scale-space function



$$L(x,\sigma) = LOG(x;\sigma) * f(x)$$

Lindeberg's Theory

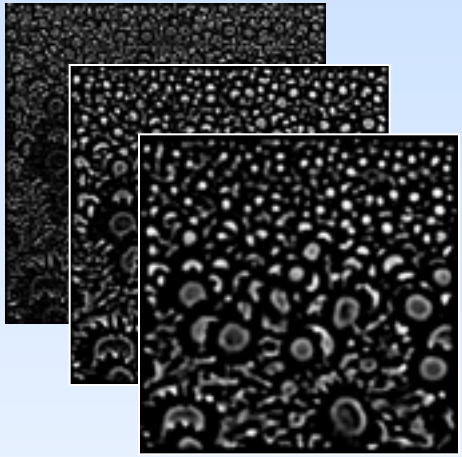
Multi-Scale Feature Selection Process

Original Image

$f(x)$

Convolve

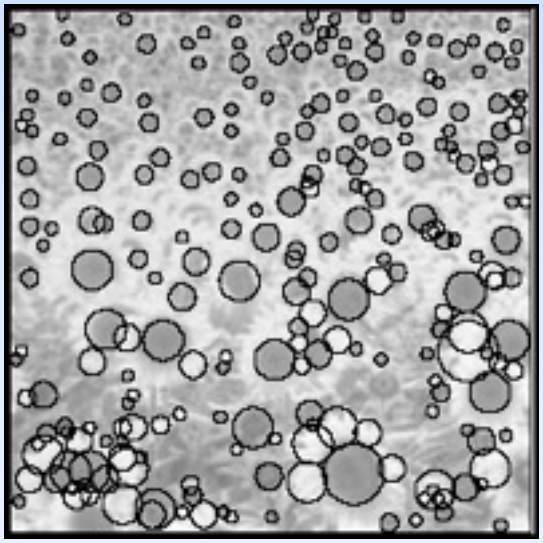
3D scale-space function



$$L(x,\sigma) = LOG(x;\sigma) * f(x)$$

Maximize

250 strongest responses
(Large circle = large scale)



Tracking Through Scale Space

Approximating LOG using DOG

$$LOG(x;\sigma) \approx DOG(x;\sigma) = G(x;\sigma) - G(x;1.6\sigma)$$

2D LOG filter with scale σ

2D DOG filter with scale σ

Tracking Through Scale Space

Approximating LOG using DOG

$$LOG(x;\sigma) \approx DOG(x;\sigma) = G(x;\sigma) - G(x;1.6\sigma)$$

2D LOG filter with scale σ

2D DOG filter with scale σ

2D Gaussian with μ=0 and scale σ

2D Gaussian with μ=0 and scale 1.6σ

Tracking Through Scale Space

Approximating LOG using DOG

$$LOG(x;\sigma) \approx DOG(x;\sigma) = G(x;\sigma) - G(x;1.6\sigma)$$

- 2D LOG filter with scale σ
- 2D DOG filter with scale σ
- 2D Gaussian with μ=0 and scale σ
- 2D Gaussian with μ=0 and scale 1.6σ

Why DOG?

- Gaussian pyramids are created faster
- Gaussian can be used as a mean-shift kernel

Tracking Through Scale Space

Approximating LOG using DOG

$$LOG(x;\sigma) \approx DOG(x;\sigma) = G(x;\sigma) - G(x;1.6\sigma)$$

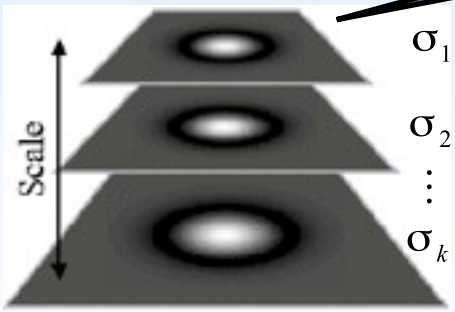
- 2D LOG filter with scale σ
- 2D DOG filter with scale σ
- 2D Gaussian with μ=0 and scale σ
- 2D Gaussian with μ=0 and scale 1.6σ

Why DOG?

- Gaussian pyramids are created faster
- Gaussian can be used as a mean-shift kernel

3D spatial kernel

$$K(x,\sigma) =$$



DOG filters at multiple scales

Tracking Through Scale Space

Approximating LOG using DOG

$$LOG(x;\sigma) \approx DOG(x;\sigma) = G(x;\sigma) - G(x;1.6\sigma)$$

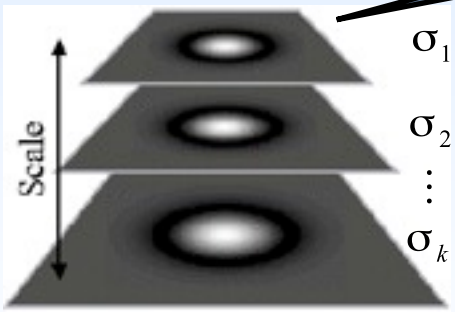
- 2D LOG filter with scale σ
- 2D DOG filter with scale σ
- 2D Gaussian with μ=0 and scale σ
- 2D Gaussian with μ=0 and scale 1.6σ

Why DOG?

- Gaussian pyramids are created faster
- Gaussian can be used as a mean-shift kernel

3D spatial kernel

$$K(x,\sigma) =$$



DOG filters at multiple scales

→ Scale-space filter bank

Tracking Through Scale Space

Using Lindeberg's Theory

Recall:

Model: $\vec{q} = (q_1, \ldots, q_m)$ at y_0

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

Color bin: $b(x)$

Pixel weight: $w(x) = \sqrt{\frac{q_{b(x)}}{p_{b(x)}(y_0)}}$

The likelihood that each candidate pixel belongs to the target

Tracking Through Scale Space

Using Lindeberg's Theory

Weight image

Recall:

Model: $\vec{q} = (q_1, \ldots, q_m)$ at y_0

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

Color bin: $b(x)$

Pixel weight: $w(x) = \sqrt{\frac{q_{b(x)}}{p_{b(x)}(y_0)}}$

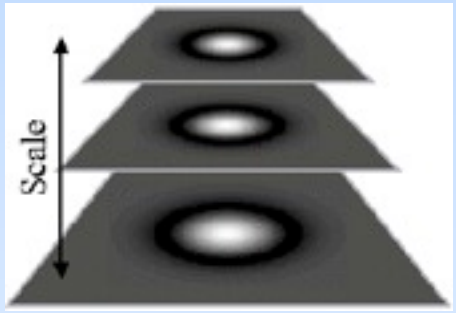
The likelihood that each candidate pixel belongs to the target

Tracking Through Scale Space

Using Lindeberg's Theory

Weight image

*



3D spatial kernel (DOG)

Centered at current location and scale

Recall:

Model: $\vec{q} = (q_1, \ldots, q_m)$ at y_0

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

Color bin: $b(x)$

Pixel weight: $w(x) = \sqrt{\frac{q_{b(x)}}{p_{b(x)}(y_0)}}$

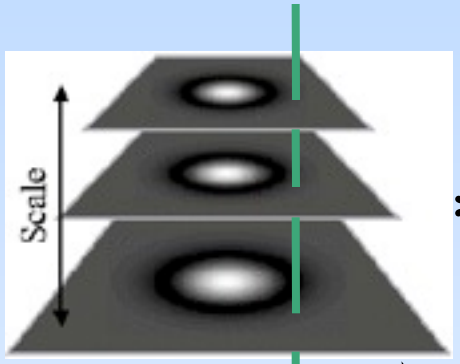
The likelihood that each candidate pixel belongs to the target

Tracking Through Scale Space

Using Lindeberg's Theory

Weight image

*



3D spatial kernel (DOG)

*

1D scale kernel (Epanechnikov)

Centered at current location and scale

Recall:

Model: $\vec{q} = (q_1, \ldots, q_m)$ at y_0

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

Color bin: $b(x)$

Pixel weight: $w(x) = \sqrt{\frac{q_{b(x)}}{p_{b(x)}(y_0)}}$

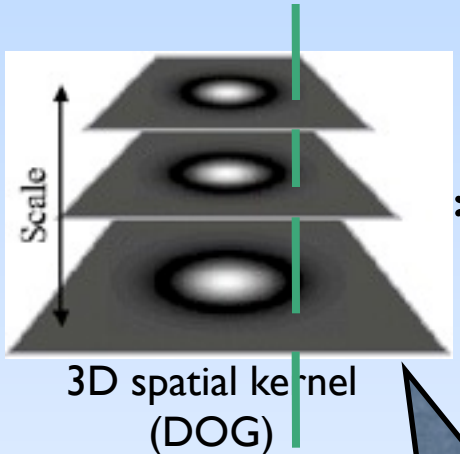
The likelihood that each candidate pixel belongs to the target

Tracking Through Scale Space

Using Lindeberg's Theory

Weight image

$*$



3D spatial kernel (DOG)

$*$

1D scale kernel (Epanechnikov)

$= E(x,\sigma)$

3D scale-space representation

Centered at current location and scale

Modes are blobs in the scale-space neighborhood

Recall:

Model: $\vec{q} = (q_1, \ldots, q_m)$ at y_0

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

Color bin: $b(x)$

Pixel weight: $w(x) = \sqrt{\frac{q_{b(x)}}{p_{b(x)}(y_0)}}$

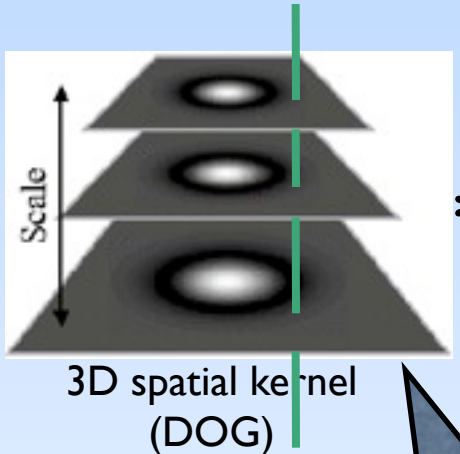
The likelihood that each candidate pixel belongs to the target

Tracking Through Scale Space

Using Lindeberg's Theory

Weight image

$*$



3D spatial kernel (DOG)

$*$

1D scale kernel (Epanechnikov)

$= E(x,\sigma)$

3D scale-space representation

Centered at current location and scale

Modes are blobs in the scale-space neighborhood

Need a mean-shift procedure that finds local modes in $E(x,\sigma)$

Recall:

Model: $\vec{q} = (q_1,\ldots,q_m)$ at y_0

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

Color bin: $b(x)$

Pixel weight: $w(x) = \sqrt{\dfrac{q_{b(x)}}{p_{b(x)}(y_0)}}$

The likelihood that each candidate pixel belongs to the target

Tracking Through Scale Space

Example

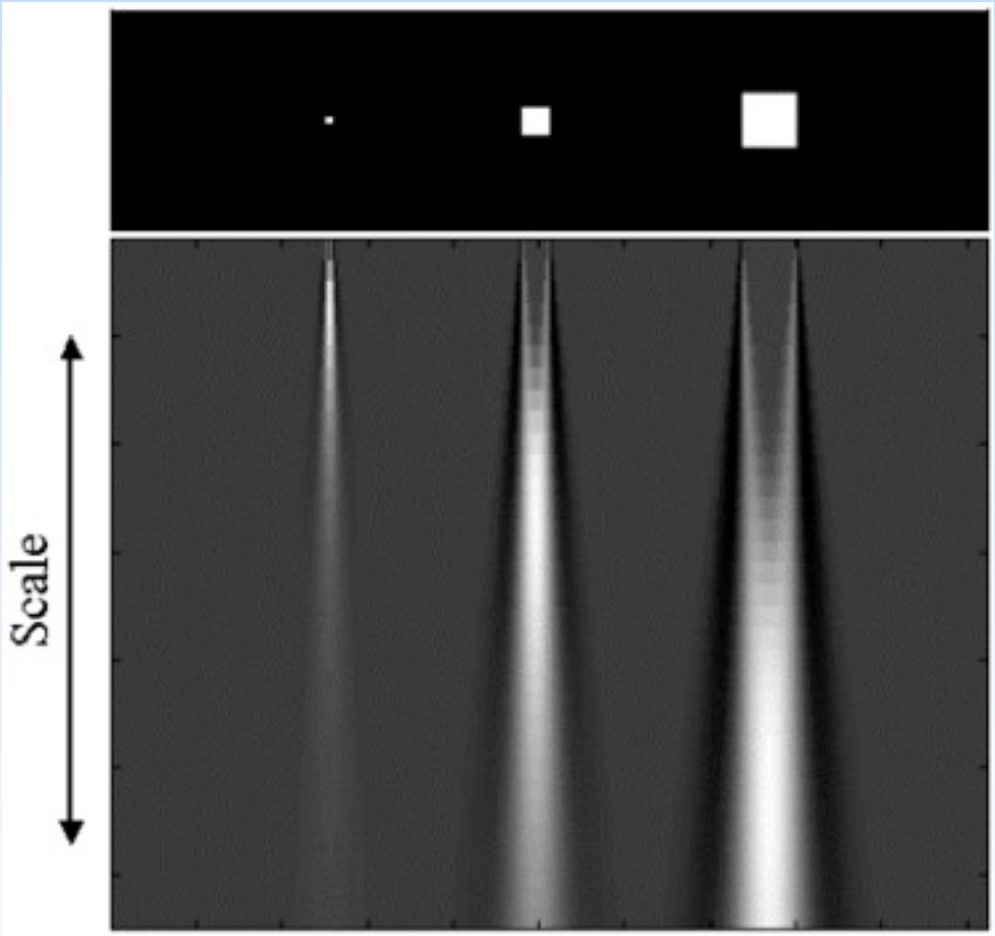


Image of 3 blobs

A slice through the 3D scale-space representation

Tracking Through Scale Space

Applying Mean-Shift

Use interleaved spatial/scale mean-shift

Spatial stage:

Fix σ and look for the best *x*

Scale stage:

Fix x and look for the best σ

Tracking Through Scale Space

Applying Mean-Shift

Use interleaved spatial/scale mean-shift

Spatial stage:

Fix σ and look for the best *x*

Scale stage:

Fix x and look for the best σ

Iterate stages until convergence of *x* and σ

Tracking Through Scale Space

Applying Mean-Shift

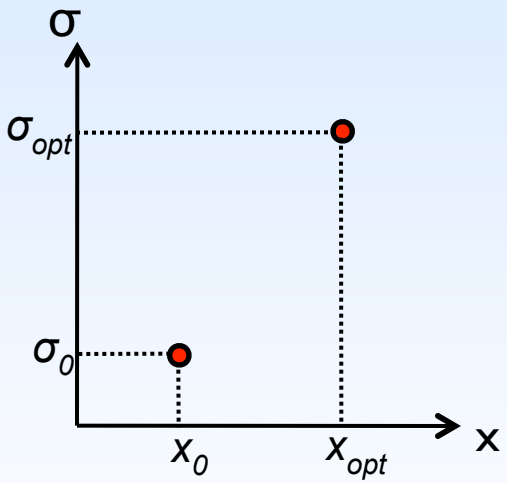
Use interleaved spatial/scale mean-shift

Spatial stage:

Fix σ and look for the best *x*

Scale stage:

Fix x and look for the best σ



Iterate stages until convergence of *x* and σ

Tracking Through Scale Space

Applying Mean-Shift

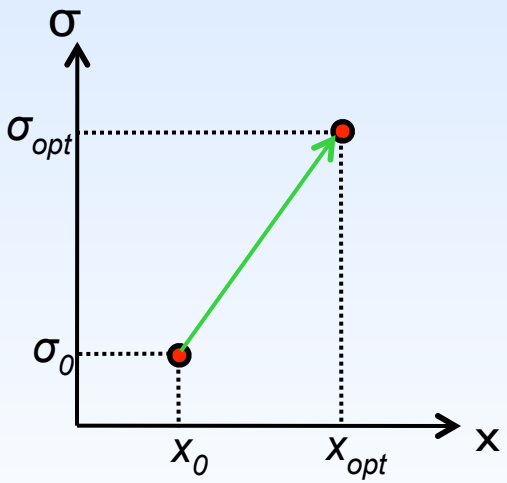
Use interleaved spatial/scale mean-shift

Spatial stage:

Fix σ and look for the best *x*

Scale stage:

Fix x and look for the best σ



Iterate stages until convergence of *x* and σ

Tracking Through Scale Space

Applying Mean-Shift

Use interleaved spatial/scale mean-shift

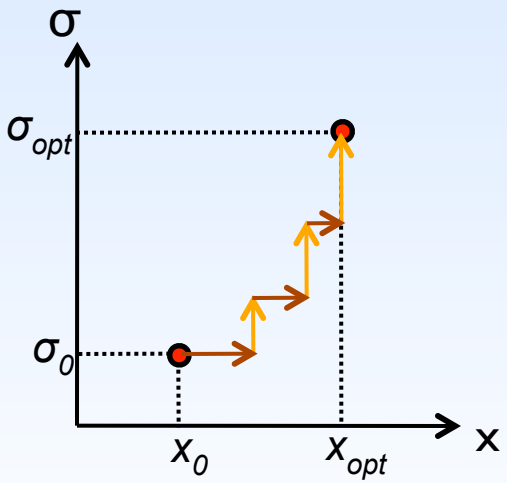
Spatial stage:

Fix σ and look for the best *x*

Scale stage:

Fix x and look for the best σ

Iterate stages until convergence of *x* and σ



Tracking Through Scale Space

Results

Fixed-scale

Tracking Through Scale Space

Results

Fixed-scale

± 10% scale adaptation

Tracking Through Scale Space

Results

Fixed-scale

± 10% scale adaptation

Tracking through scale space