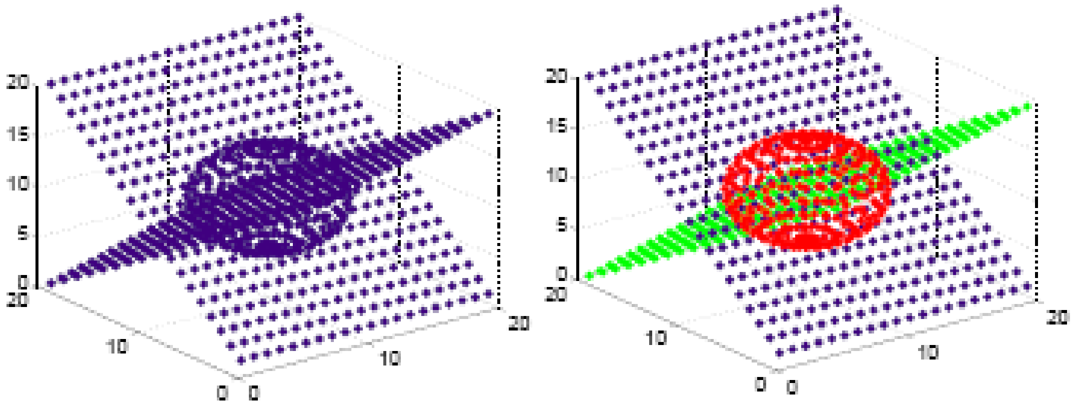
Mean Shift: theory and applications

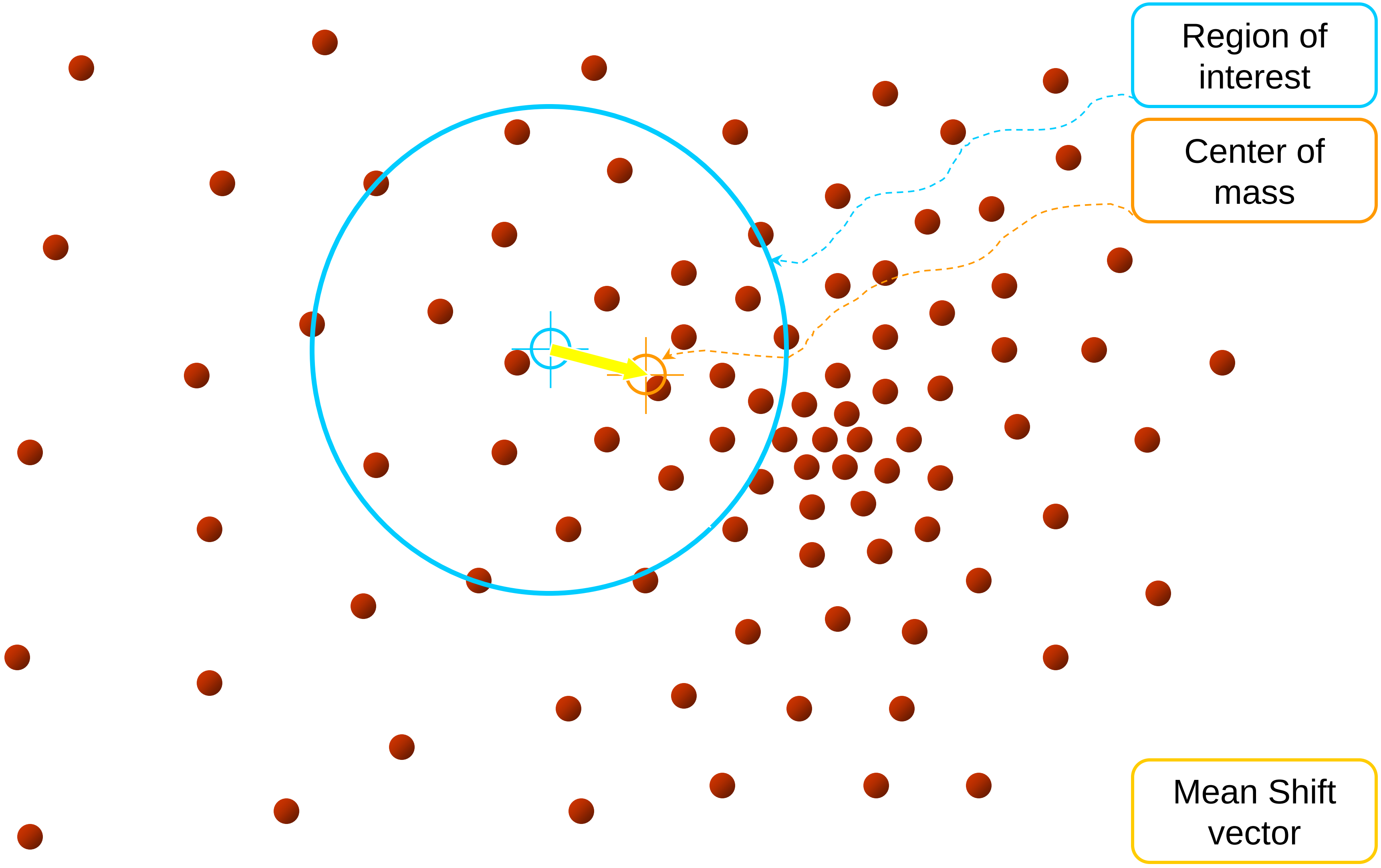


Summary

- **Fundamentals**
  - Basic Idea
  - Preliminaries: Parzen Windows
  - Mean Shift
    - Introduction
    - Properties
- **Applications**
  - Clustering
  - Discontinuity Preserving Smoothing
  - 2D Segmentation
  - N-D Segmentation
    - Geometrical data, Biomedical data

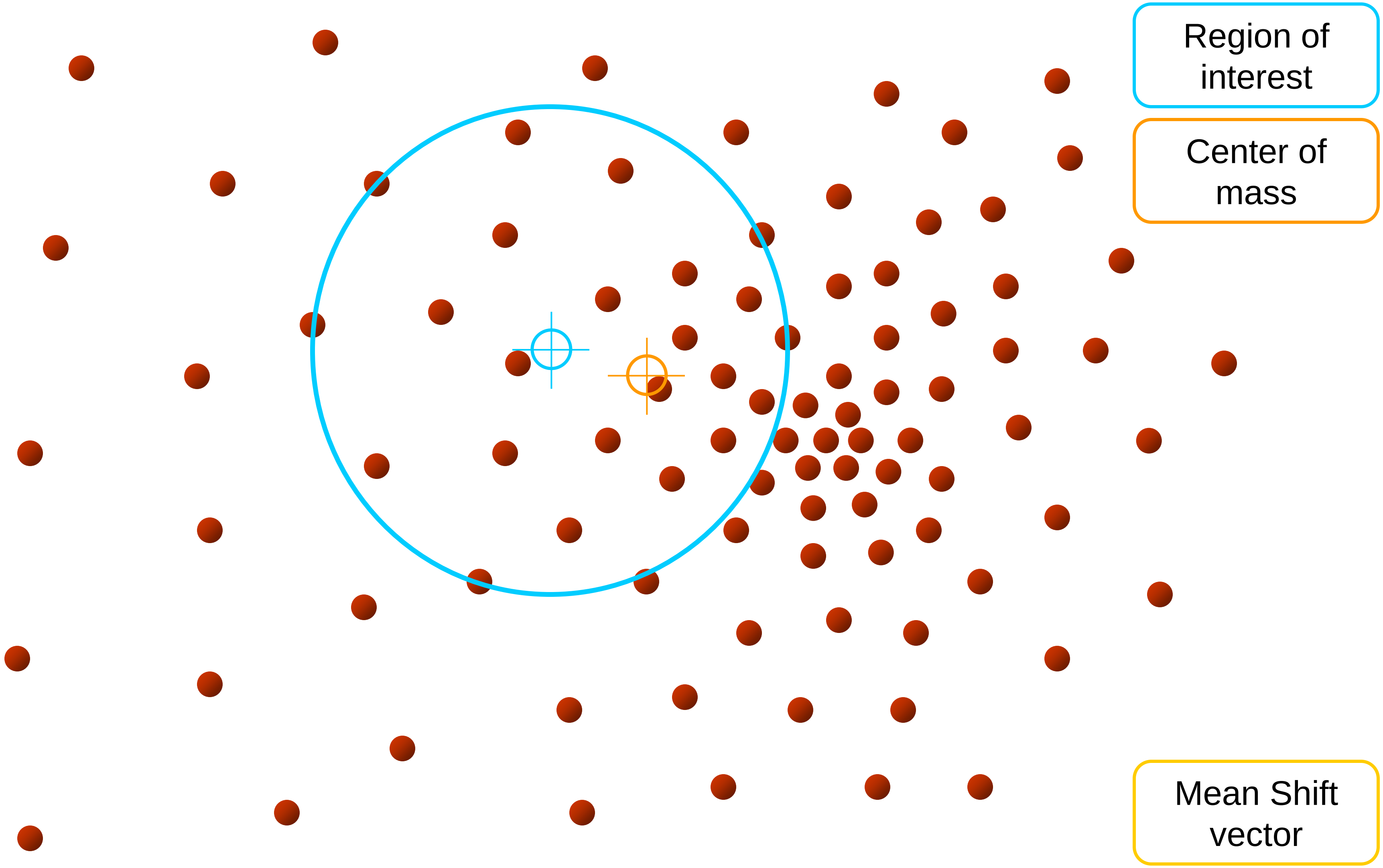
Fundamentals

Intuitive Description



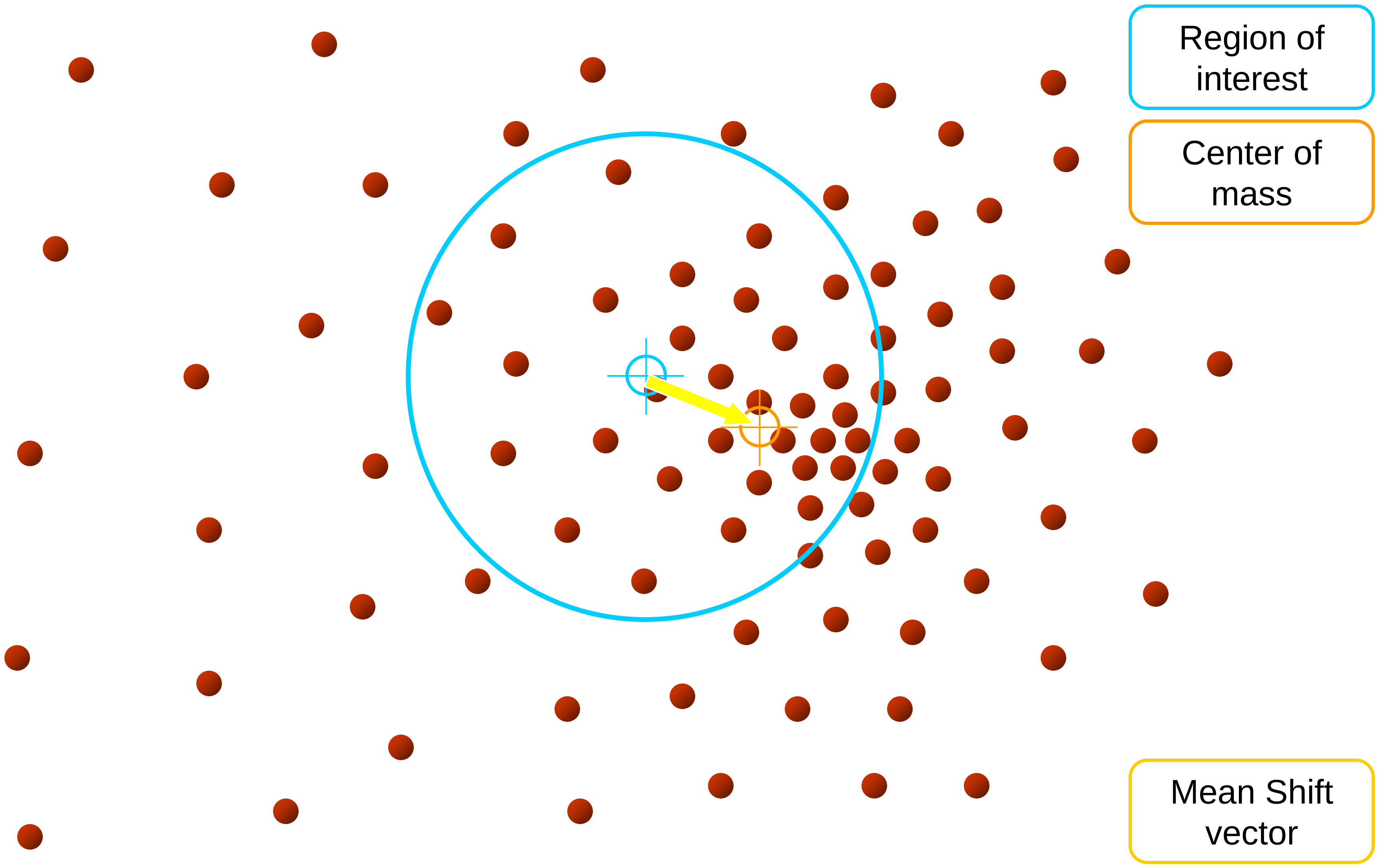
Objective : Given a set of points, find the densest region

Intuitive Description



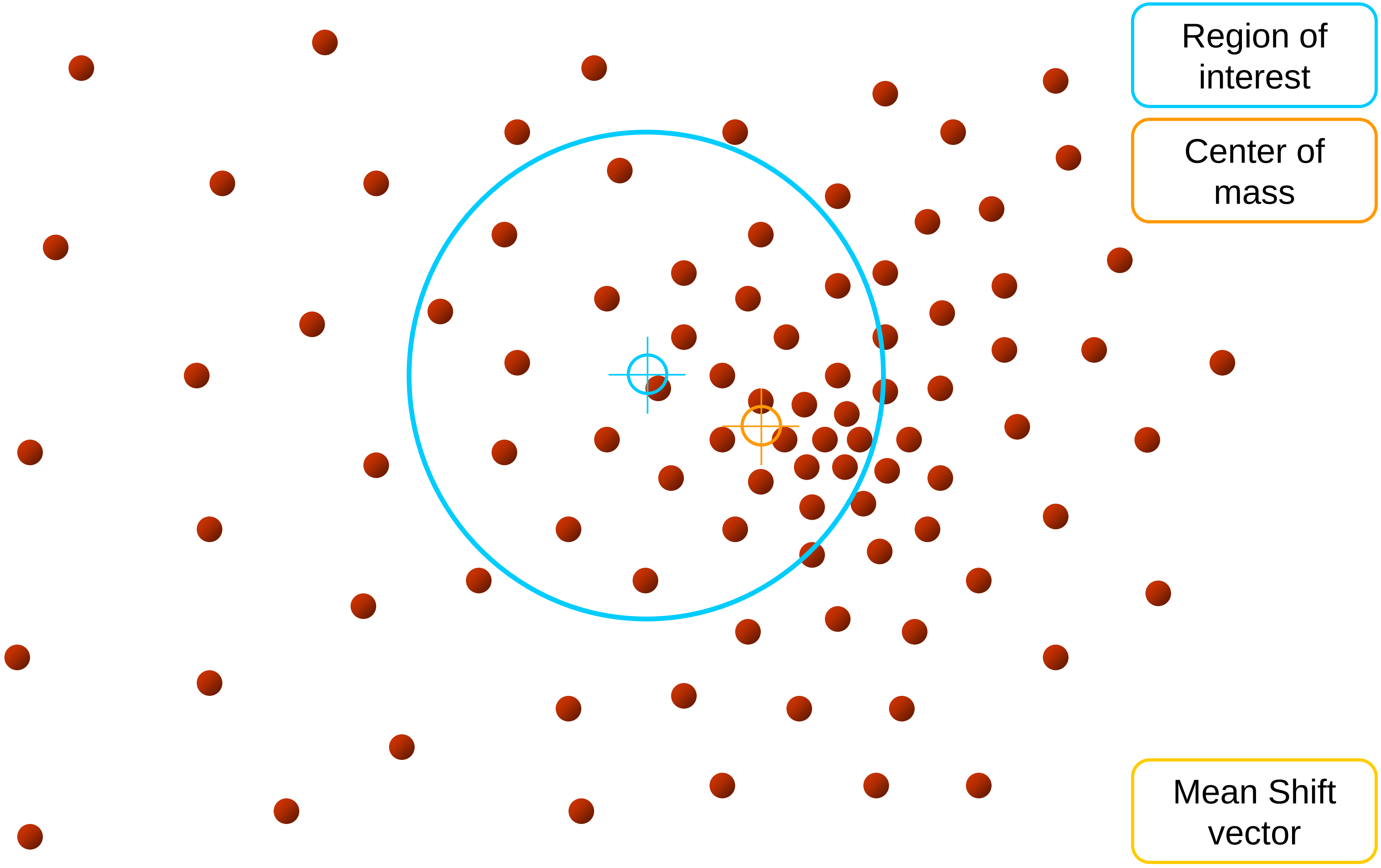
<u>Objective</u> : Given a set of points, find the densest region

Intuitive Description



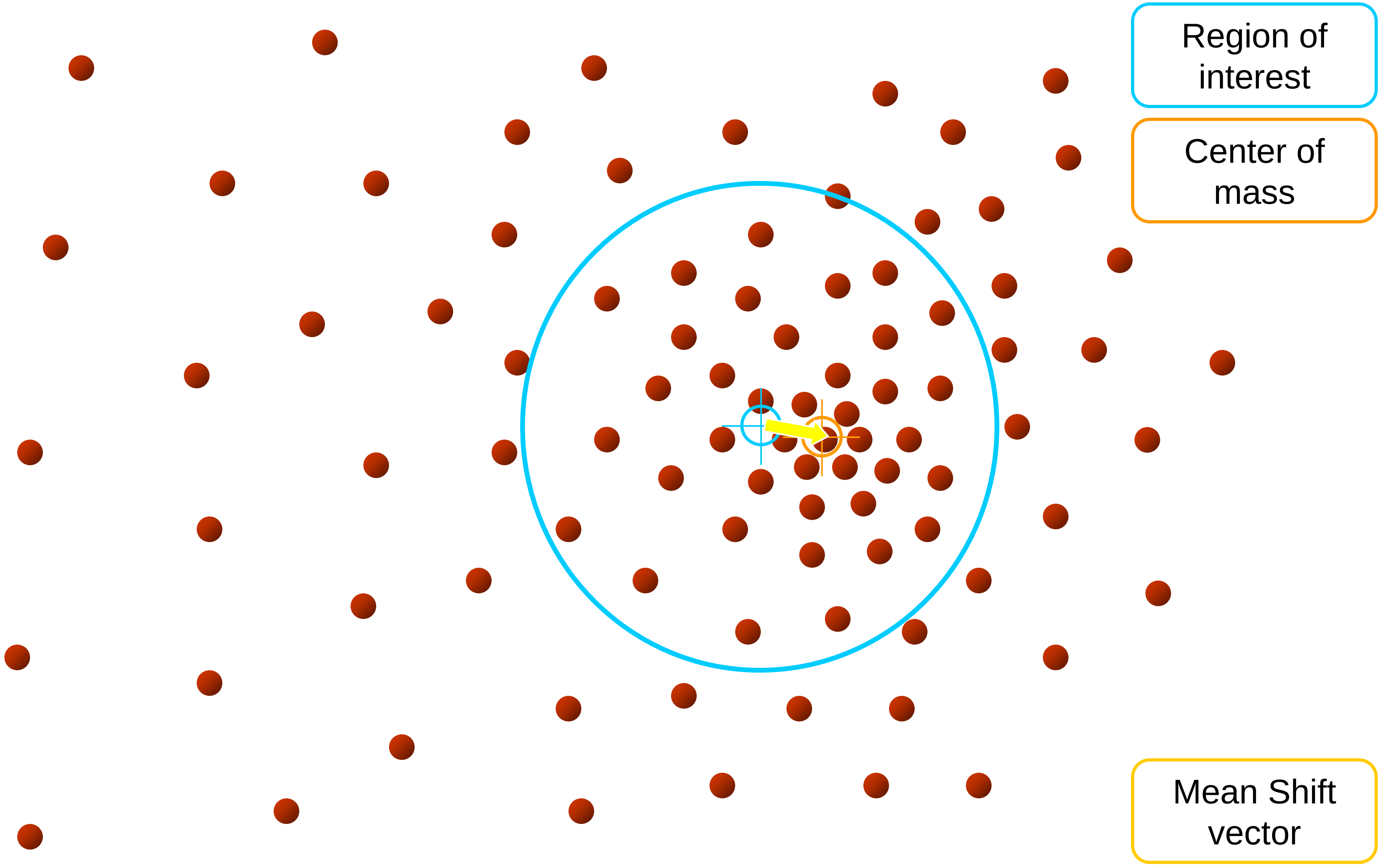
Objective : Given a set of points, find the densest region

Intuitive Description



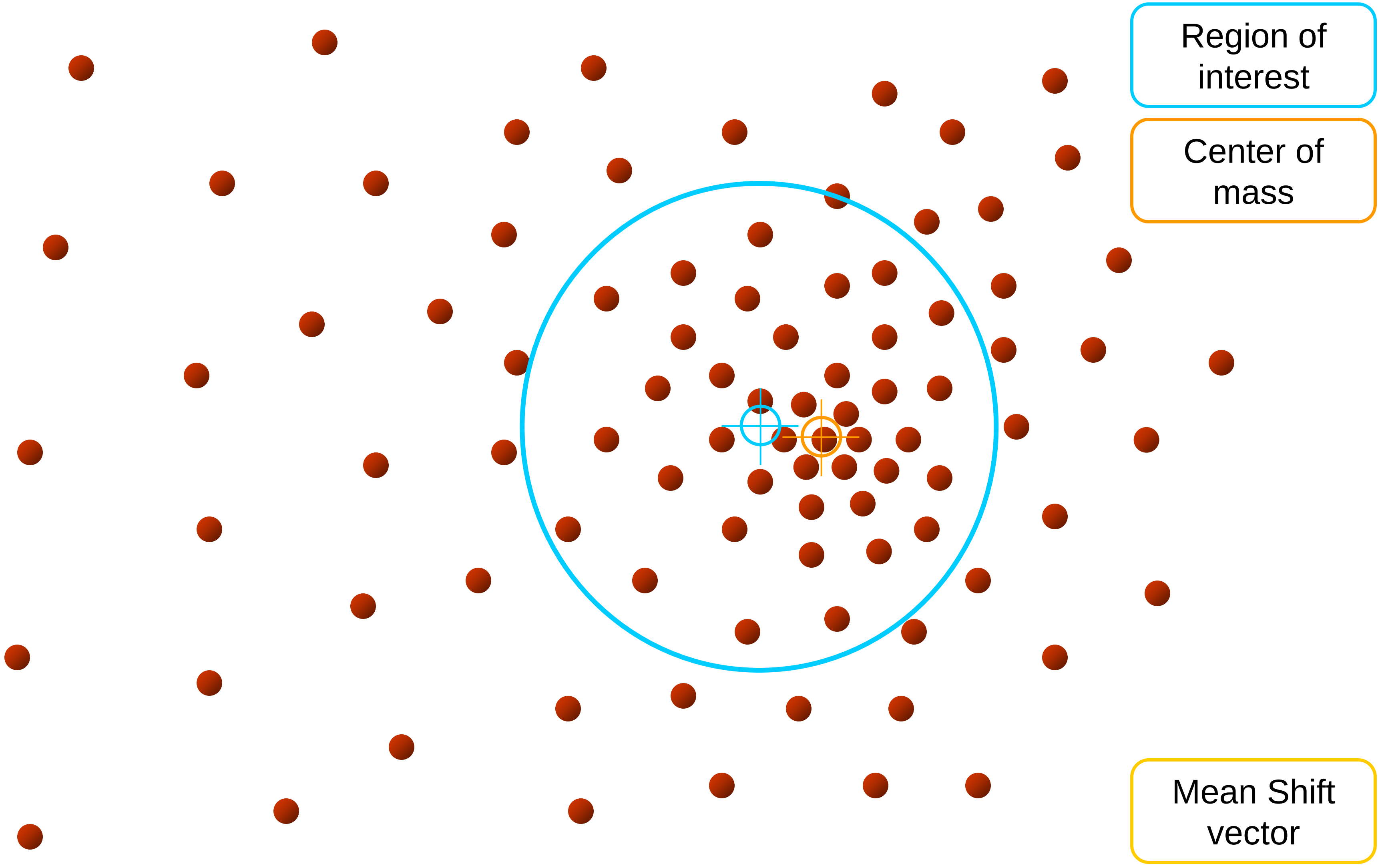
<u>Objective</u> : Given a set of points, find the densest region

Intuitive Description



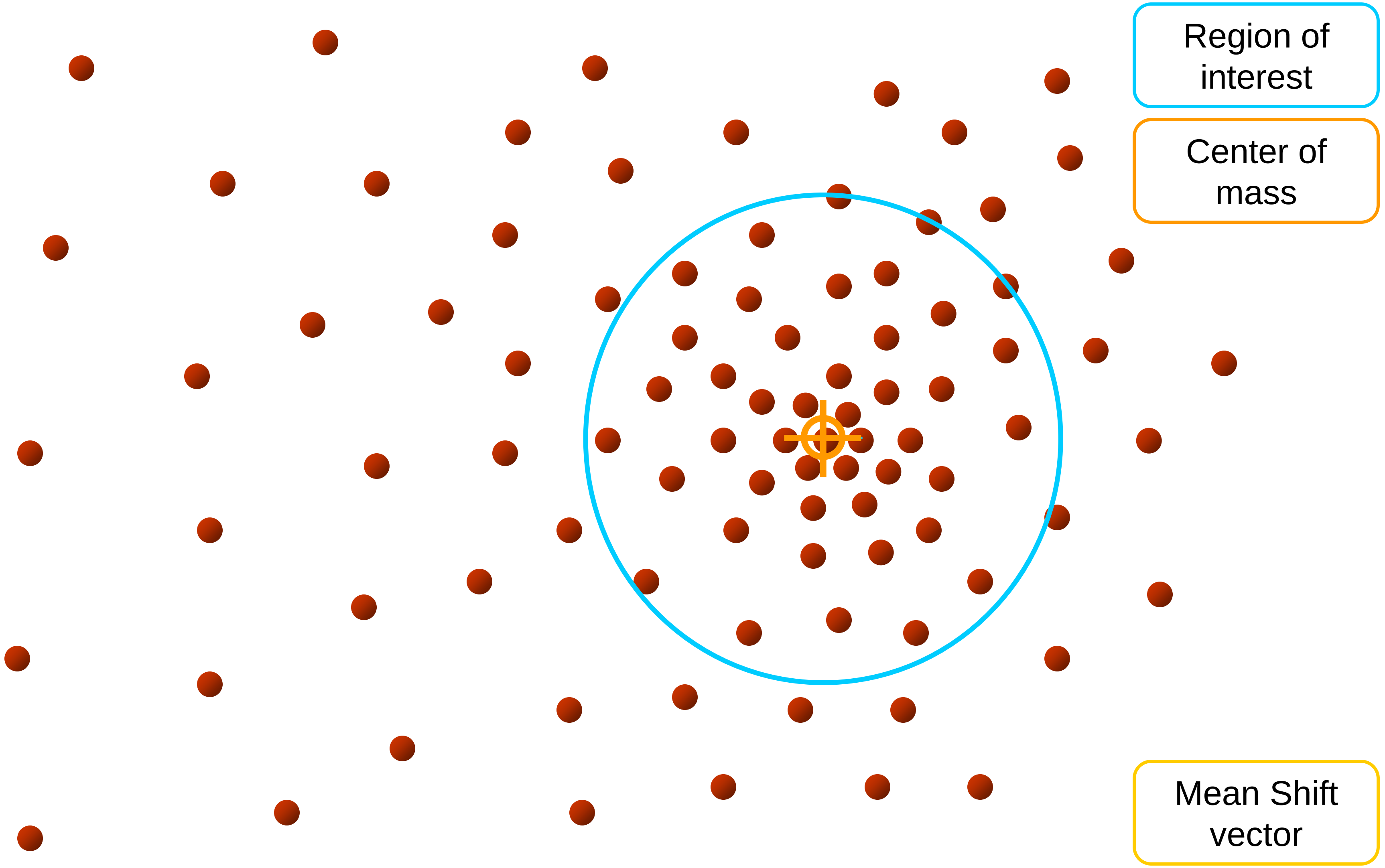
<u>Objective</u> : Given a set of points, find the densest region

Intuitive Description



Objective : Given a set of points, find the densest region

Intuitive Description



<u>Objective</u> : Given a set of points, find the densest region

What is Mean Shift ?

- A technique for *finding* modes in a set of data samples, manifesting an underlying probability density function (PDF) in R^N

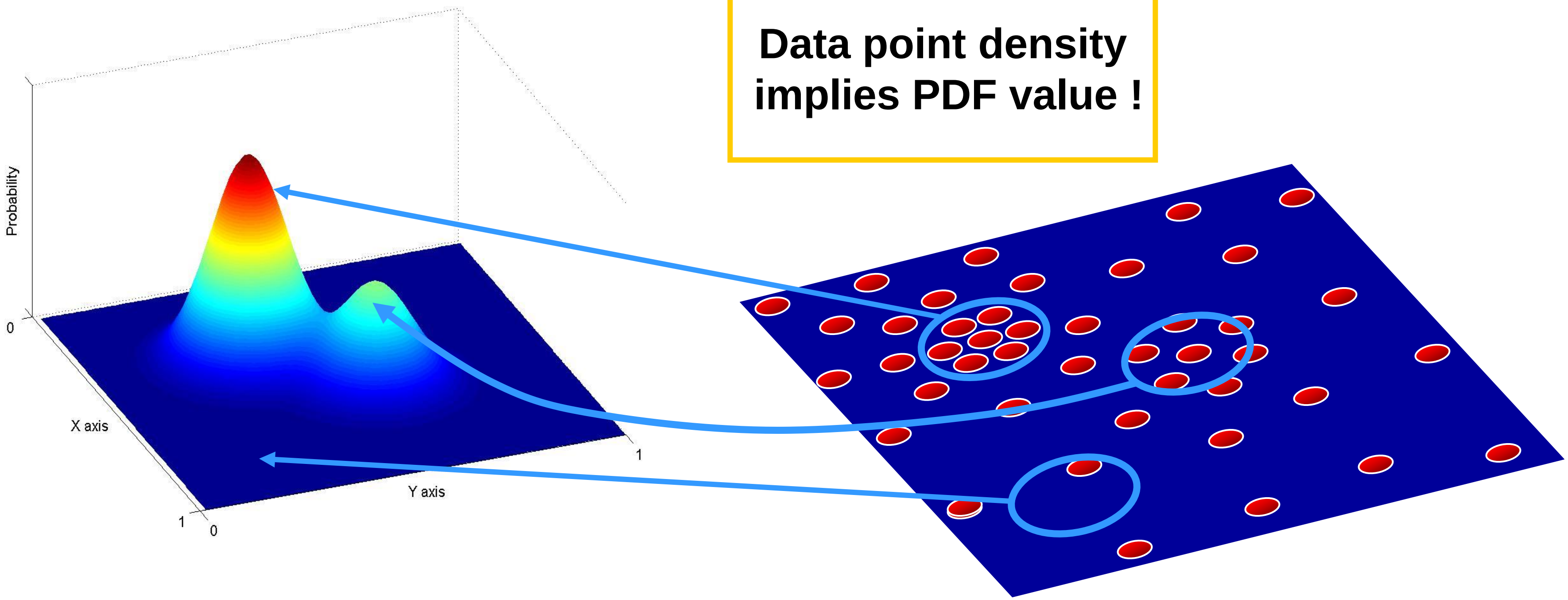
- The samples (and the related PDF) can represent and characterize different objects features:
  - Position
  - Color
  - …

Preliminaries: Parzen Windows

Non-Parametric Density Estimation

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

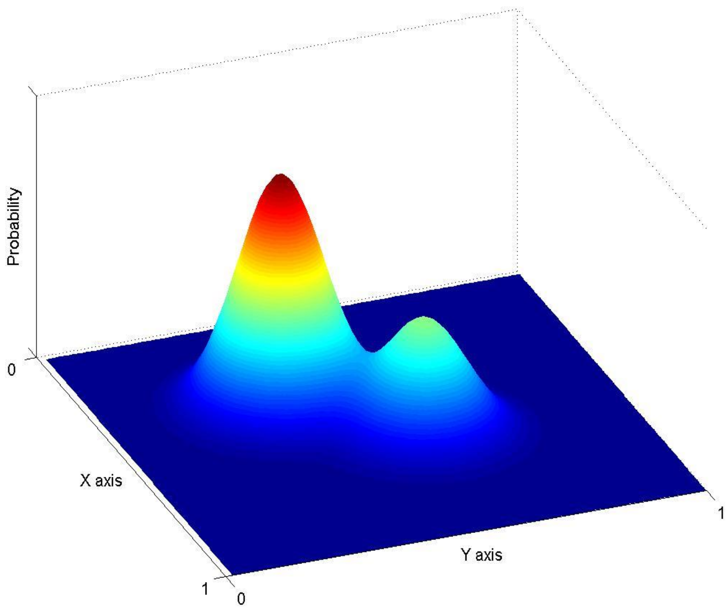
Data point density implies PDF value !



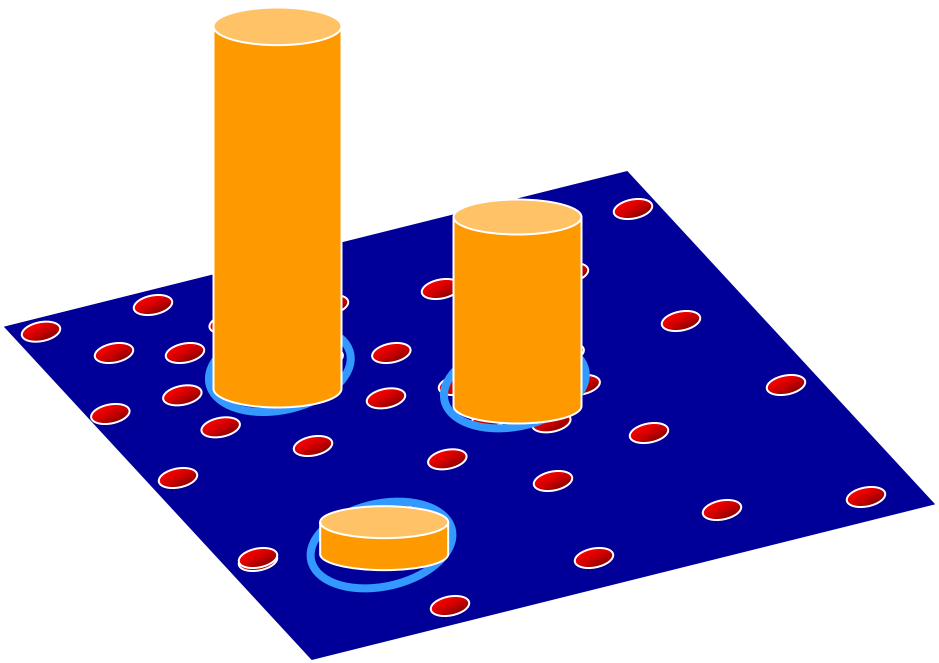
Assumed Underlying PDF

Real Data Samples

Non-Parametric Density Estimation

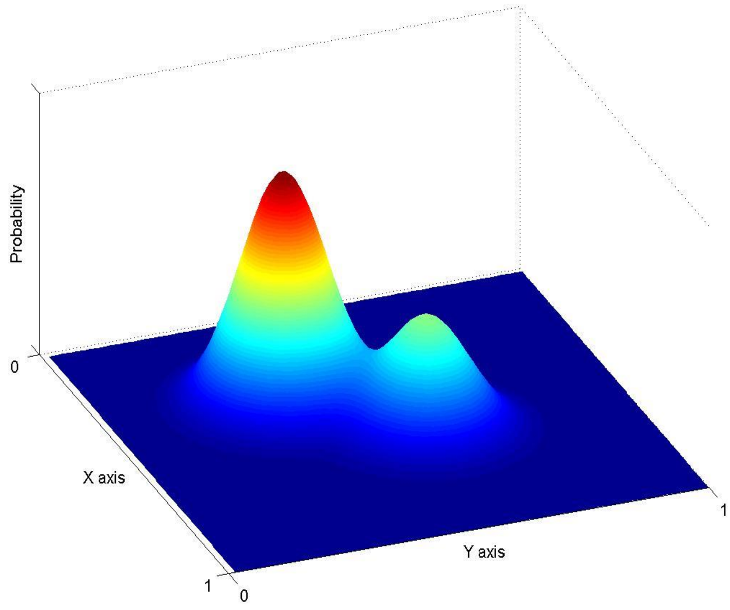


Assumed Underlying PDF

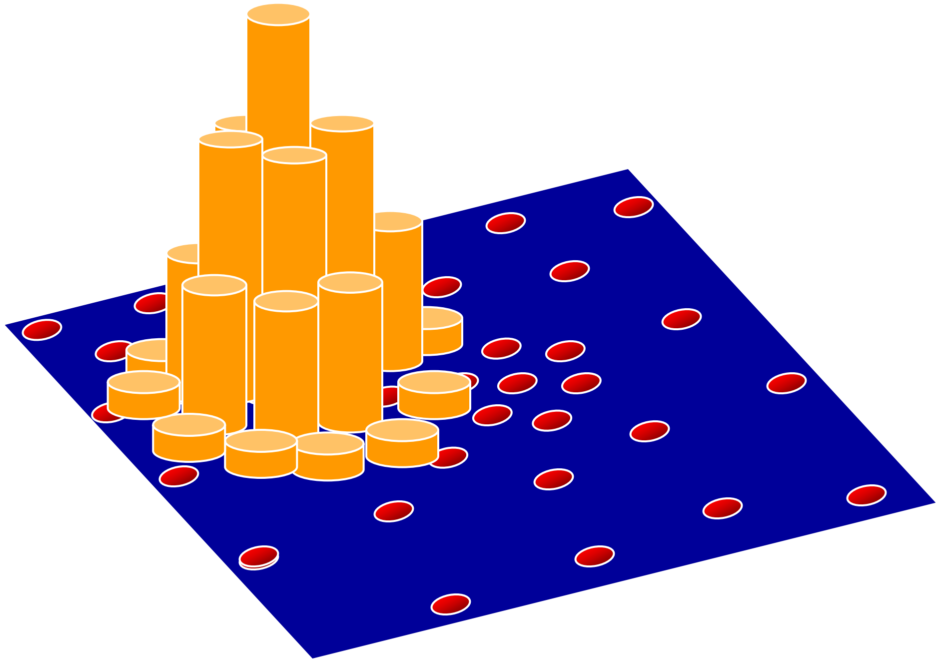


Real Data Samples

Non-Parametric Density Estimation



Assumed Underlying PDF



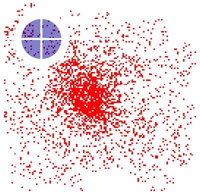
Real Data Samples

Kernel Density Estimation

Parzen Windows - General Framework

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

Kernel K(·): function of some finite number of data points $x_1 \ldots x_n$



Kernel Properties:

- See the Parzen Windows properties…

Kernel Density Estimation

Parzen Windows - Function Forms

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

Kernel K(·): function of some finite number of data points $x_1 \ldots x_n$

In practice one uses the forms:

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

Same function on each dimension

Function of vector length only

The 1D function *k* is called ***profile*** of the kernel

Kernel Density Estimation

Various Kernels

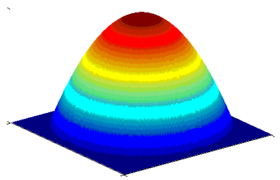
$$P(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n} K(\mathbf{x} - \mathbf{x}_i)$$

Kernel K(·): function of some finite number of data points $x_1 \ldots x_n$

Examples:

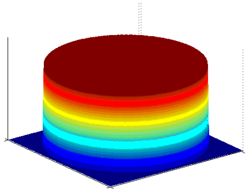
- *Epanechnikov Kernel*

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



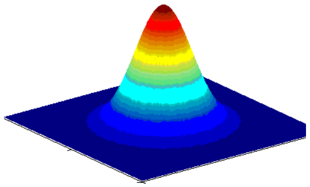
- *Uniform Kernel*

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



- *Normal Kernel*

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



Mean Shift

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 **<u>ONLY</u>** the gradient

Using the Kernel form:

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

Size of window

We get :

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

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

Kernel Density Estimation / Computing The Mean Shift

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

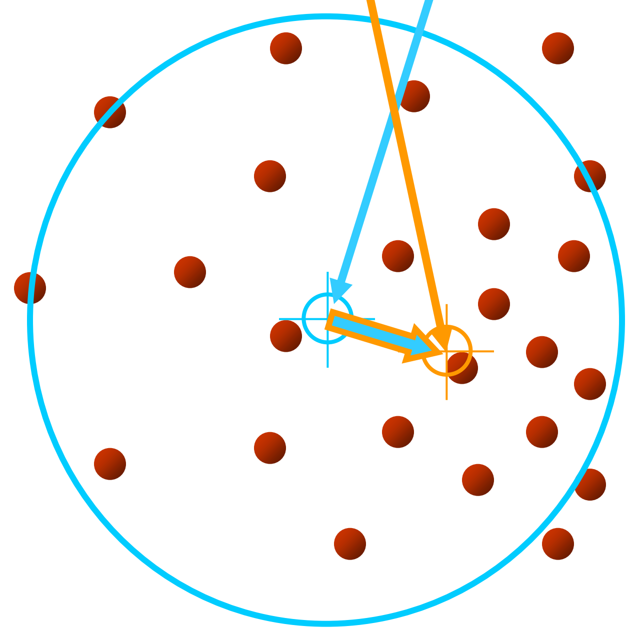
$$\nabla P(\mathbf{x}) = \frac{c}{n}\sum_{i=1}^{n} \nabla k_i = \frac{c}{n}\left[\sum_{i=1}^{n} g_i\right]\left[\frac{\sum_{i=1}^{n} \mathbf{x}_i g_i}{\sum_{i=1}^{n} g_i} - \mathbf{x}\right]$$

Computing The Mean Shift

$$\frac{c}{n}\left[\sum_{i=1}^{n} g_i\right]$$

$$\left[\frac{\sum_{i=1}^{n} \mathbf{x}_i g_i}{\sum_{i=1}^{n} g_i} - \mathbf{x}\right]$$

Yet another Kernel
density estimation !



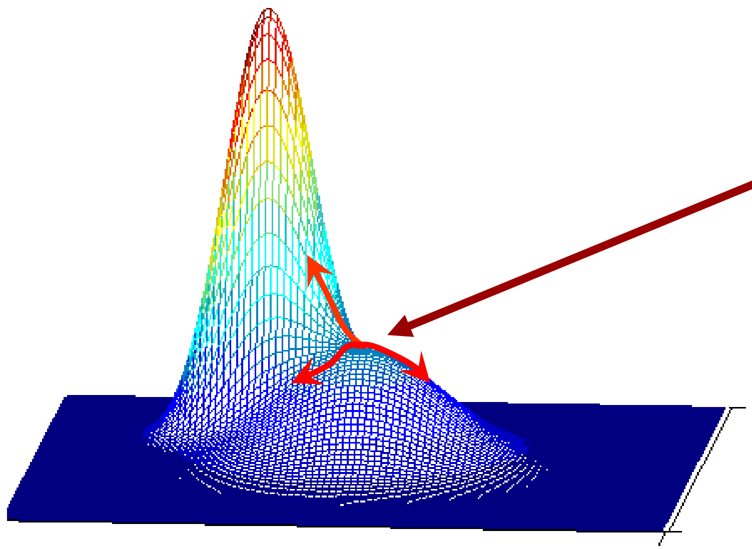
Simple Mean Shift procedure:

- Compute Mean Shift vector

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

- Translate the Kernel window by **m(x)** until convergence (m(x)<thresh)

Mean Shift Mode Detection



What happens if we reach a saddle point

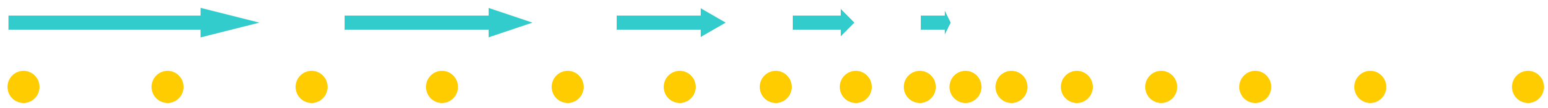
?

Perturb the mode position and check if we return back

<u>Updated Mean Shift Procedure:</u>

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

Mean Shift Properties

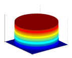


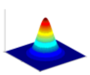
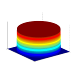
- **Automatic convergence speed** – the Mean Shift vector size depends on the gradient itself.

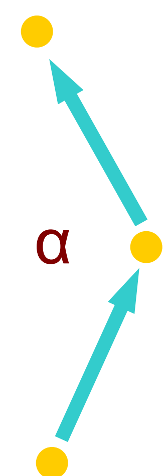
- Near maxima, the steps are small and refined

Adaptive Gradient Ascent

- Convergence is guaranteed for infinitesimal steps only ➔ **infinitely convergent** (therefore set a lower bound on the minimal distance covered after a step) [Comaniciu 2002, Chong 1995].

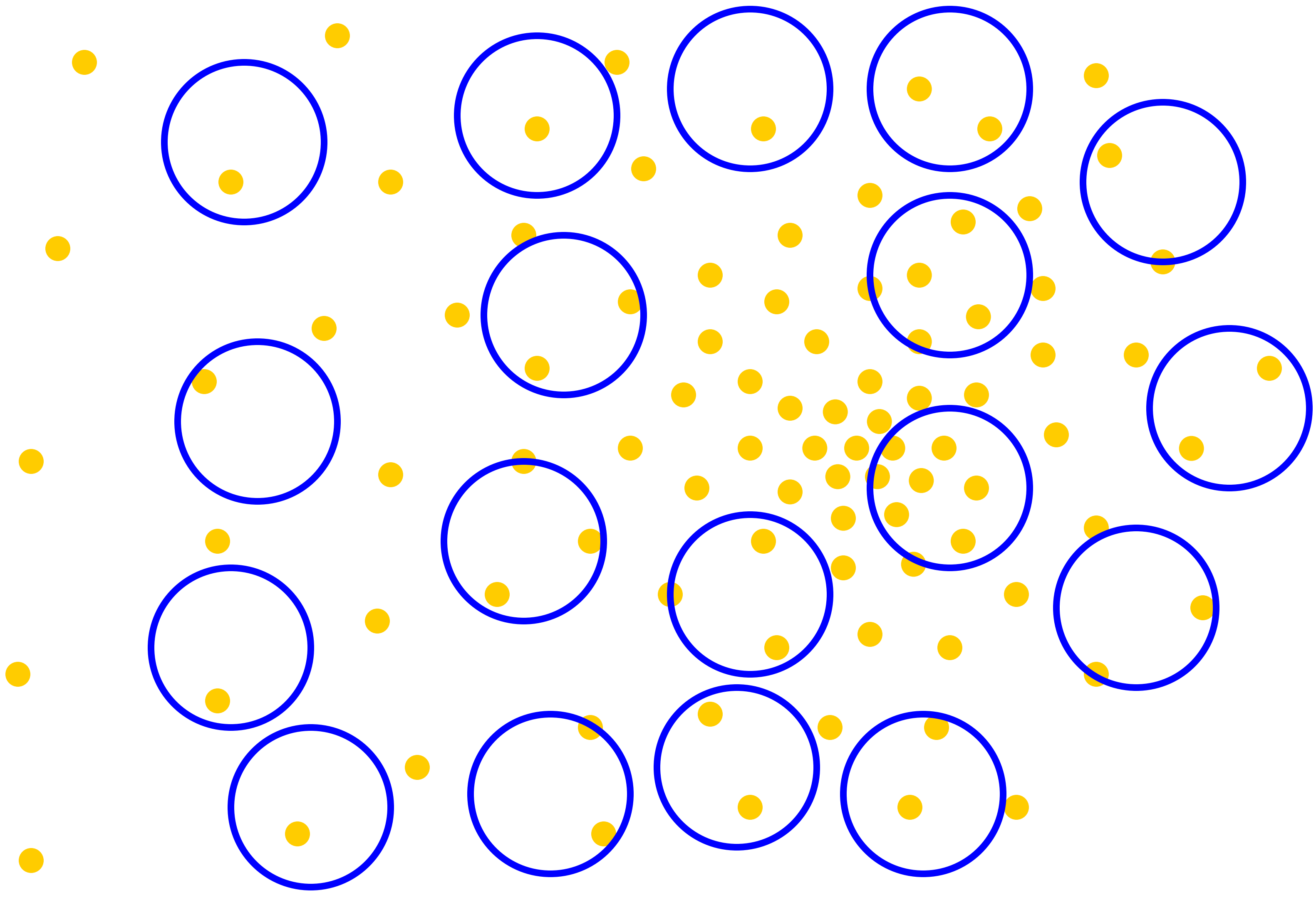
- For *Uniform Kernel* (), **convergence is achieved in a finite number of steps** [Comaniciu 2002].

- *Normal Kernel* () **exhibits a smooth trajectory, but is slower than Uniform Kernel** () [Comaniciu 2002].



$90^\circ < \alpha < 135^\circ$

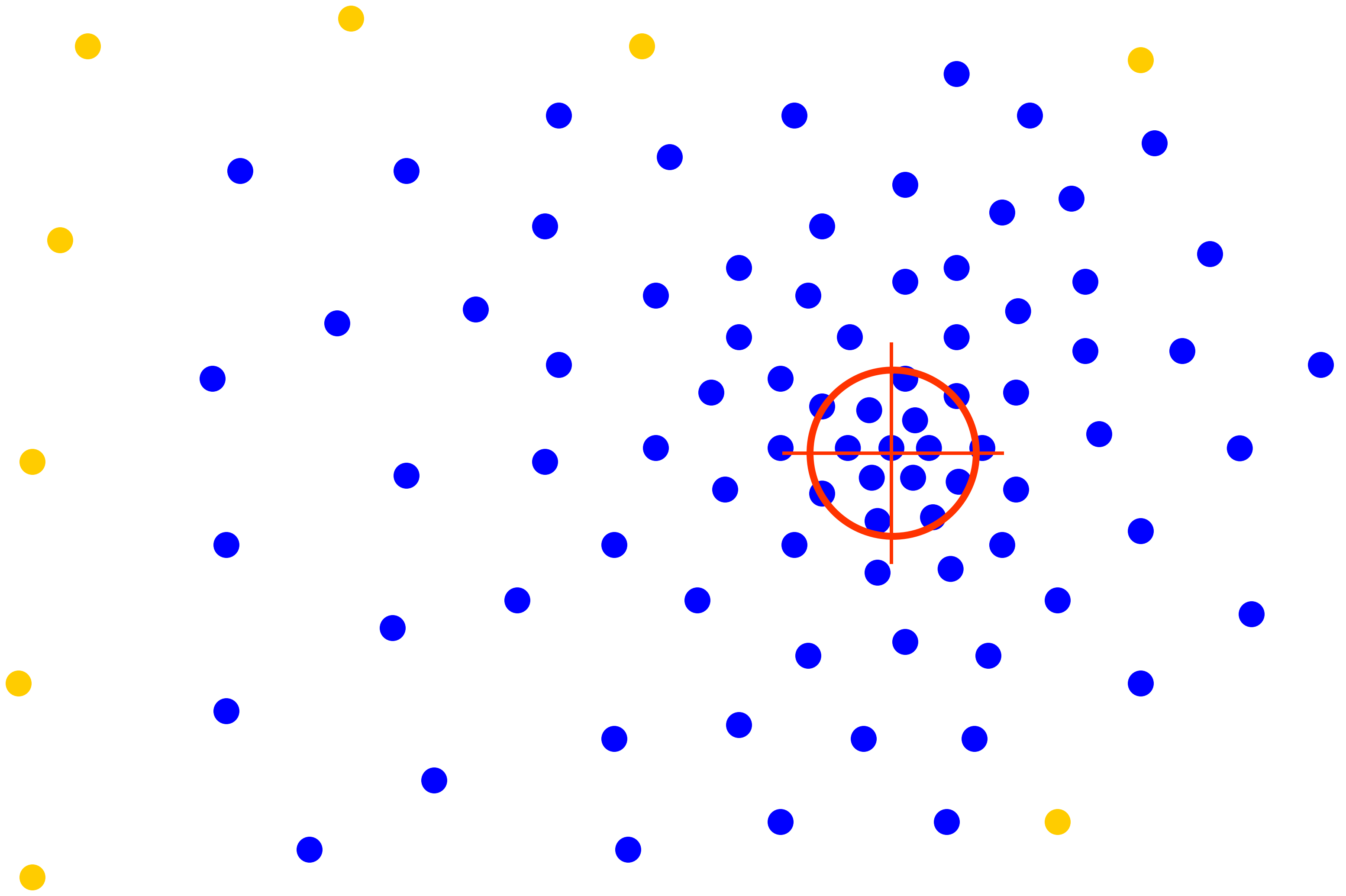
Facts - Real Modality Analysis



Tessellate the space with windows

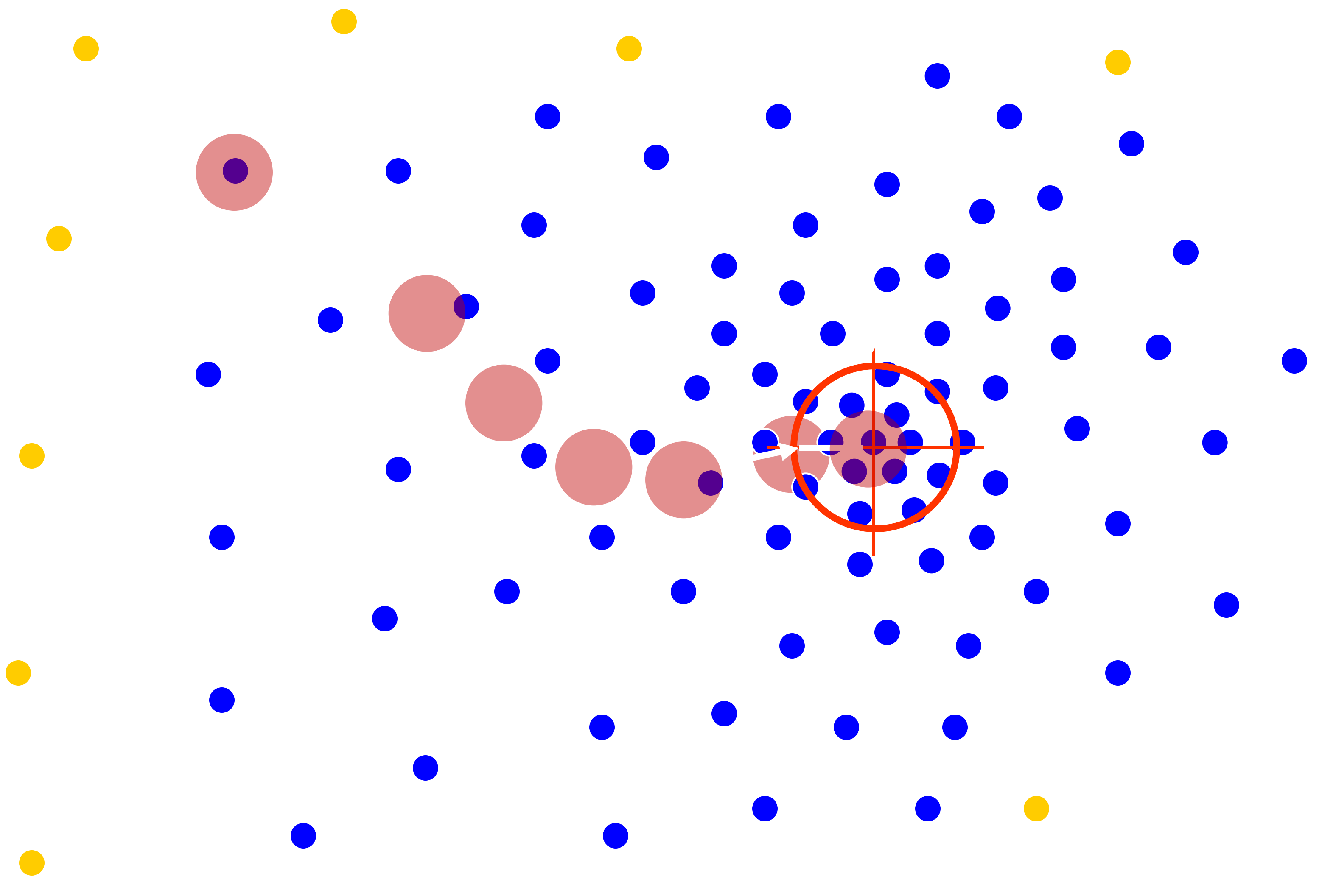
Run the procedure in parallel

Facts - Real Modality Analysis



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

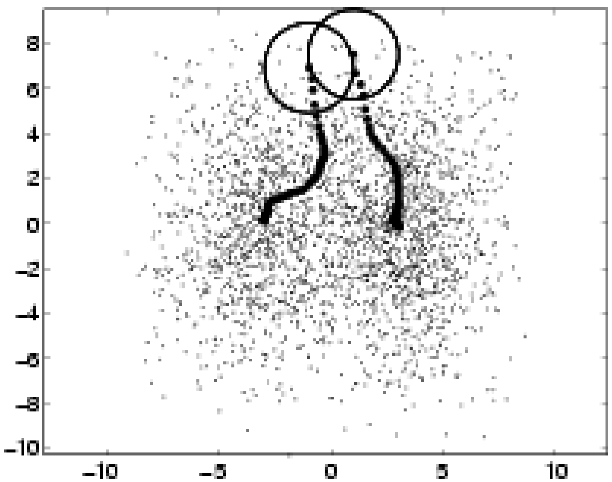
Facts - Data Analysis



Each point x_i generates a trajectory formed by $y_1 \ldots y_C$

Real Modality Analysis

An example

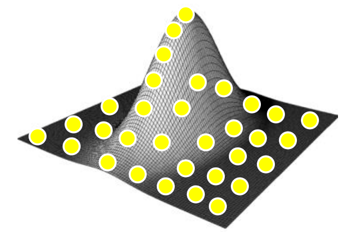


Window tracks signify the steepest ascent directions

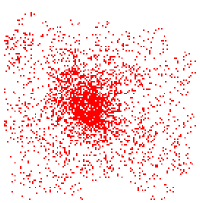
Remarks - Parzen Windows vs Mean Shift

Parzen Windows

Non-parametric
Density Estimation



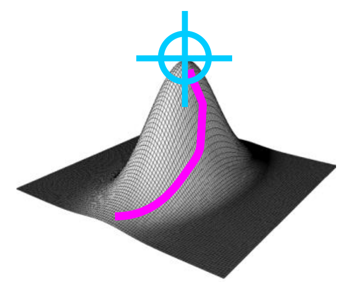
Discrete PDF Representation



Data

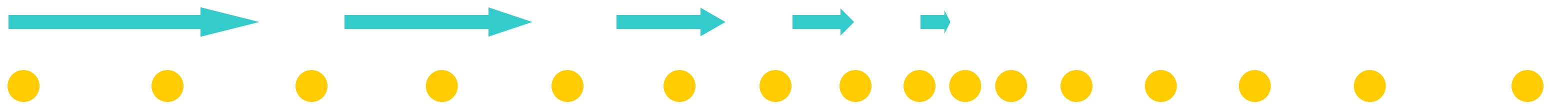
Mean Shift

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



PDF Analysis

Mean Shift Strengths & Weaknesses



Strengths :

- Application independent technique
- 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)**

Weaknesses :

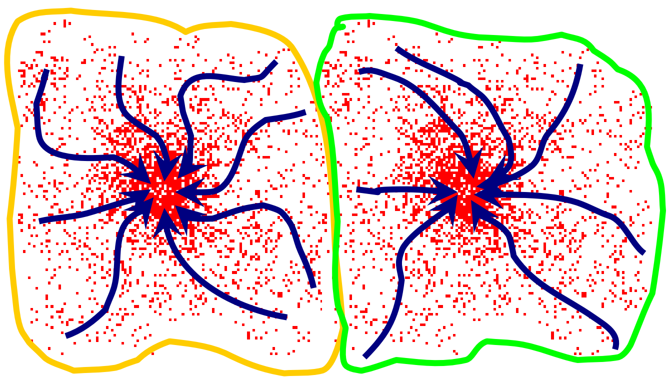
- The window size (bandwidth selection) is not trivial
  - Inappropriate window size can cause modes to be merged, or generate additional "shallow" modes ➔ **Use adaptive window size**

Mean Shift applications: Clustering

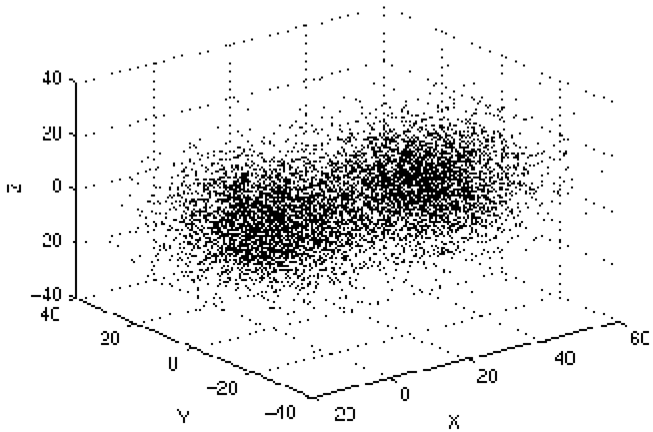
Clustering

<u>Cluster</u> : All data points in the ***attraction basin*** of a mode

<u>Attraction basin</u> : the region for which all trajectories lead to the same mode



Clustering
Synthetic Examples



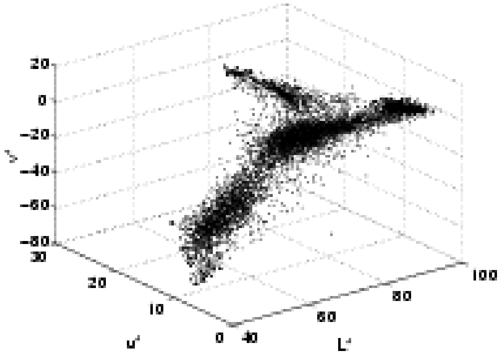
Simple Modal Structures

Complex Modal Structures

Clustering

Real Example

Feature space:
L*u*v representation



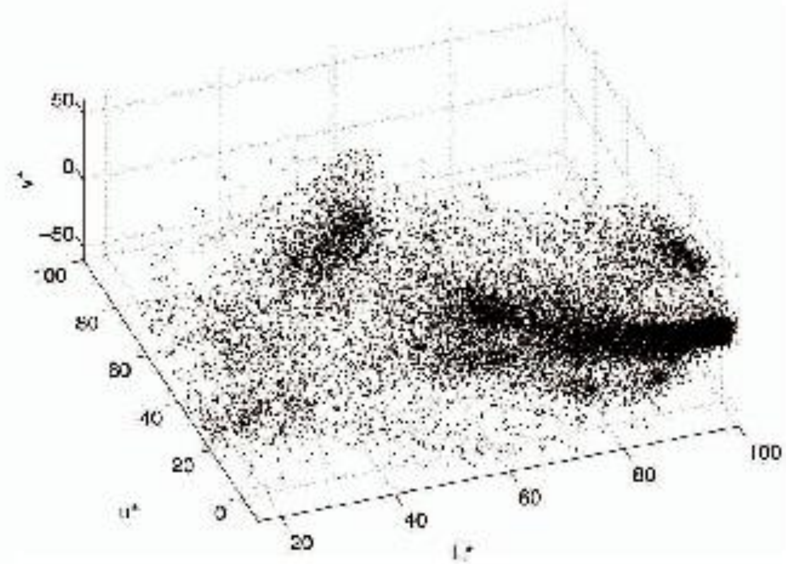
(a)

Initial window centers

Modes found

Modes after pruning

Clustering
Real Example



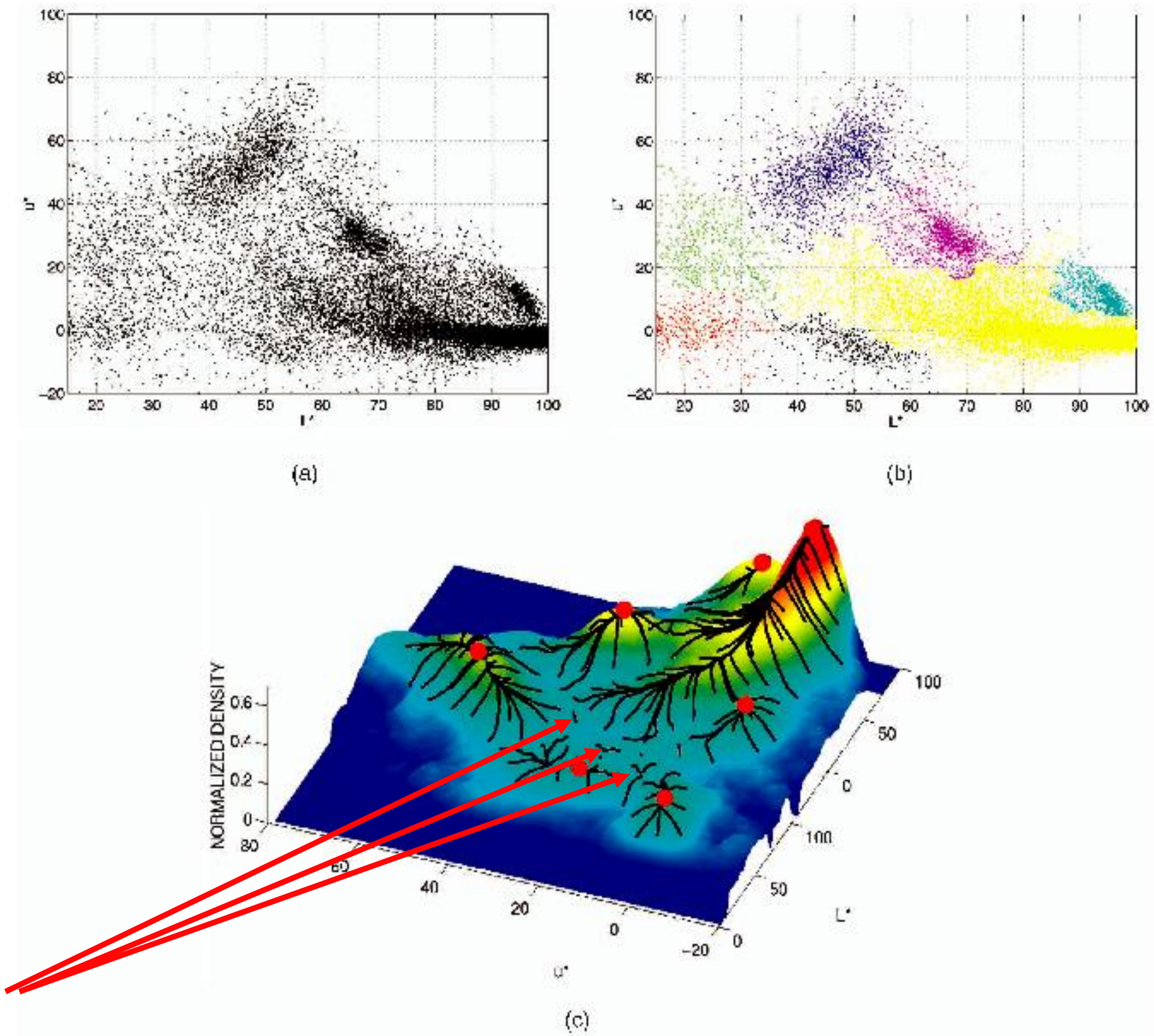
L*u*v space representation

Clustering

Real Example

2D (L*u) space representation

Final clusters



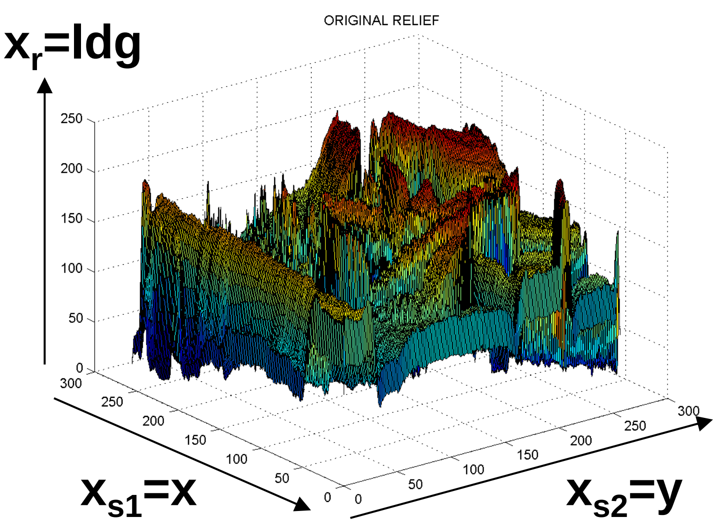
From the attraction basin points depart and reach different modes

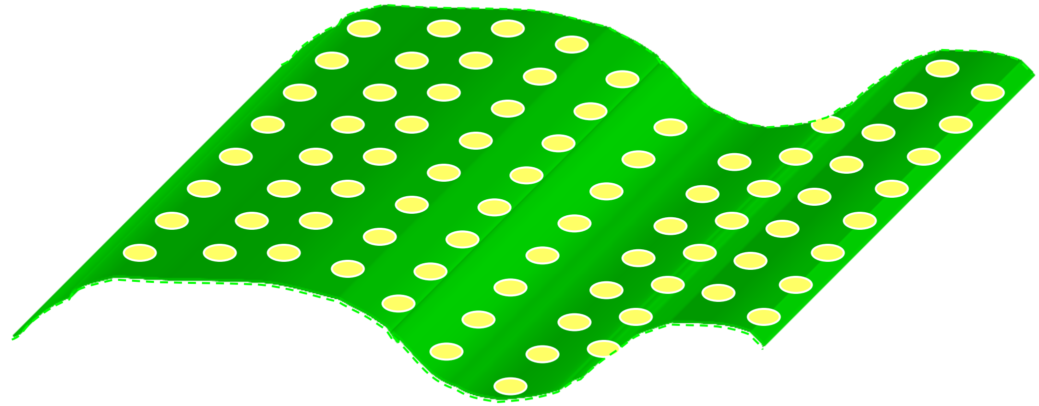
Mean Shift applications:

Discontinuity Preserving Smoothing

Discontinuity Preserving Smoothing

The image gray levels…
… can be viewed as data points in the x_s, x_r space (joined *spatial* And *color* space)





Discontinuity Preserving Smoothing

Feature space : Joint domain = spatial coordinates + color space

$$K(\mathbf{x}) = C \cdot k_s\left(\left\|\frac{\mathbf{x}^s}{h_s}\right\|\right) \cdot k_r\left(\left\|\frac{\mathbf{x}^r}{h_r}\right\|\right)$$

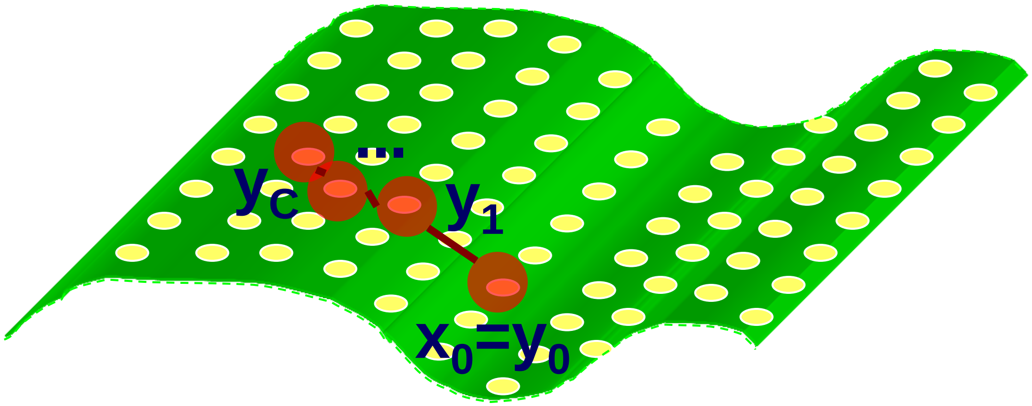
Meaning : treat the image as data points in the spatial and gray level domain

Discontinuity Preserving Smoothing

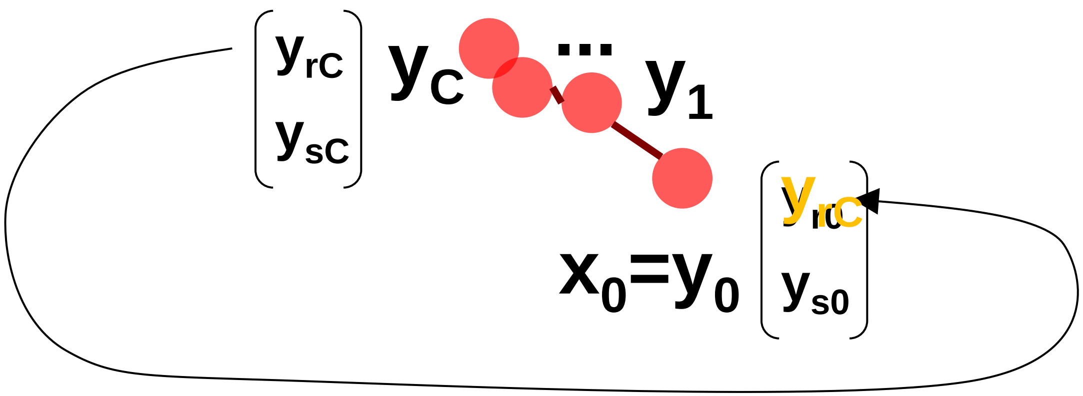
Algorithm:

1) **For each pixel**, run the MS procedure generating in the joint *spatial-chromatic* domain a trajectory

$$\mathbf{x_0 = y_0, y_1, \ldots, y_C}$$

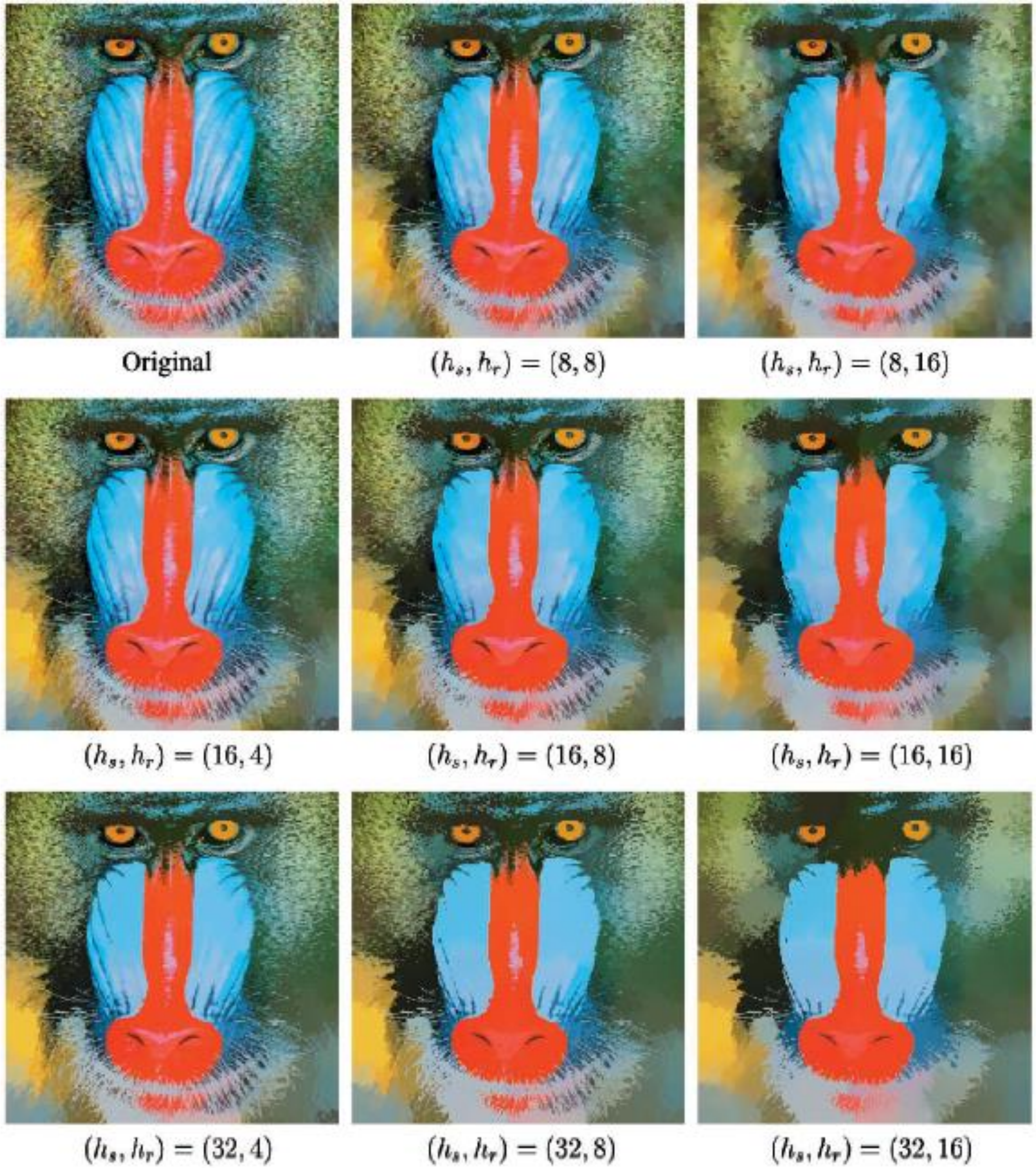


2) assign **to each pixel** the gray level of the mode reached



Discontinuity Preserving Smoothing

The effect of window size in spatial and range spaces



Discontinuity Preserving Smoothing

Example

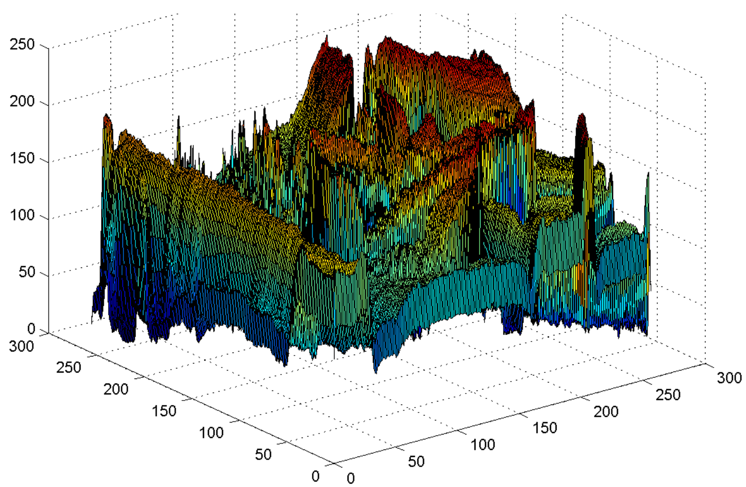
Original

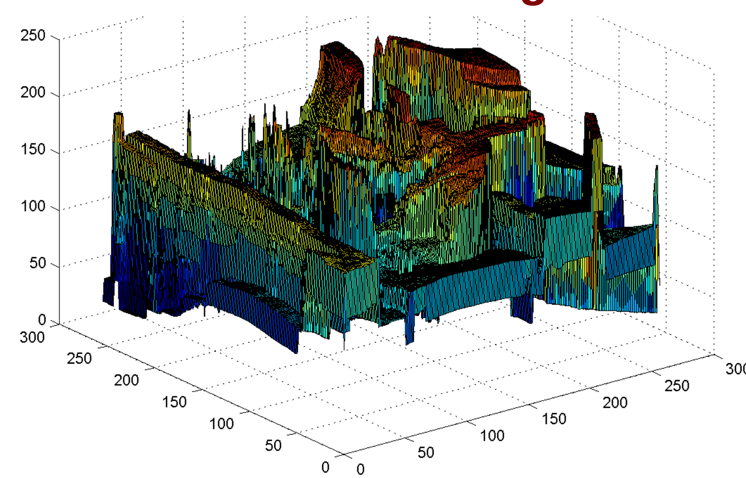
After smoothing

Discontinuity Preserving Smoothing

Example

Original | **After smoothing**





Mean Shift applications:
2D Segmentation

Segmentation

Algorithm:

• Run Filtering (*discontinuity preserving smoothing*)

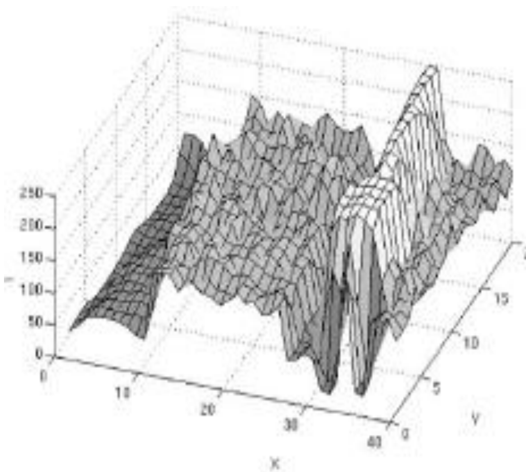
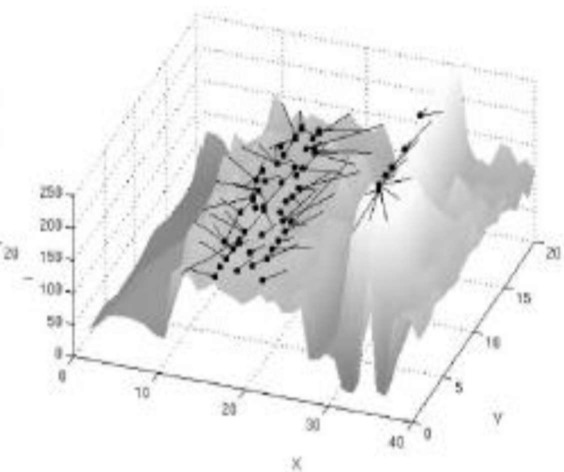
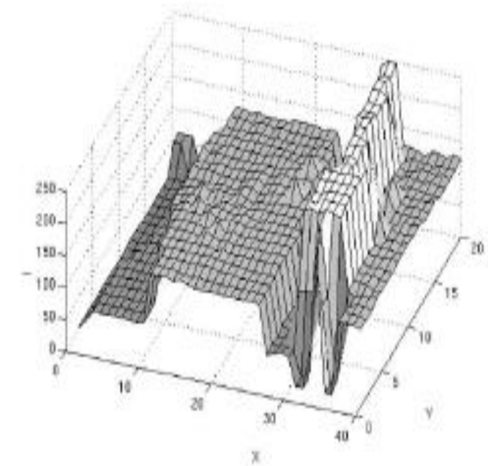


Image Data (slice)

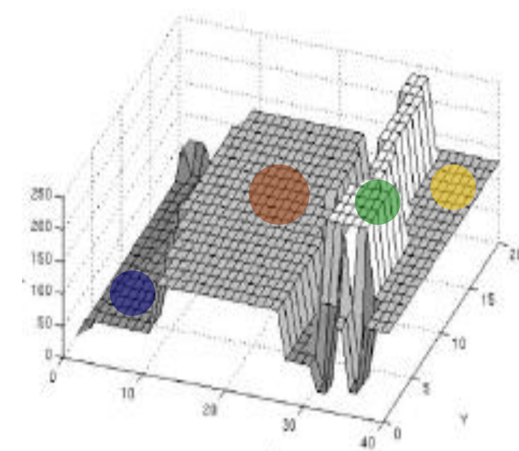


Mean Shift vectors



Smoothing result

• Cluster the clusters which are closer than window size



Segmentation result

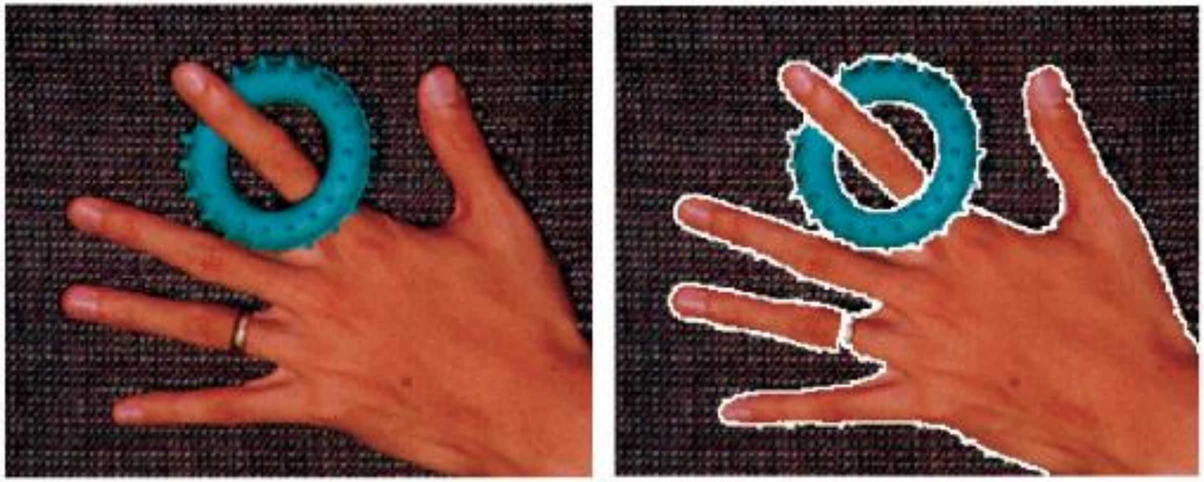
Segmentation

Example

Segmentation
Example

Segmentation

Example



Segmentation

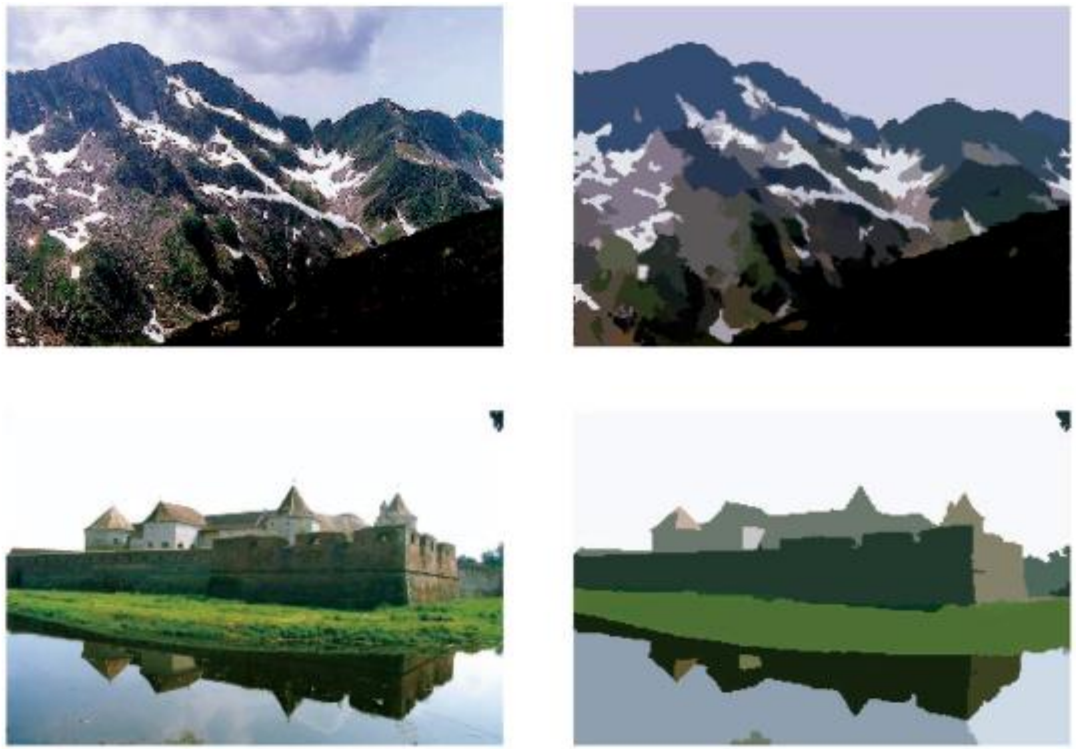
Example

Segmentation
Example

…when feature space is only gray levels…

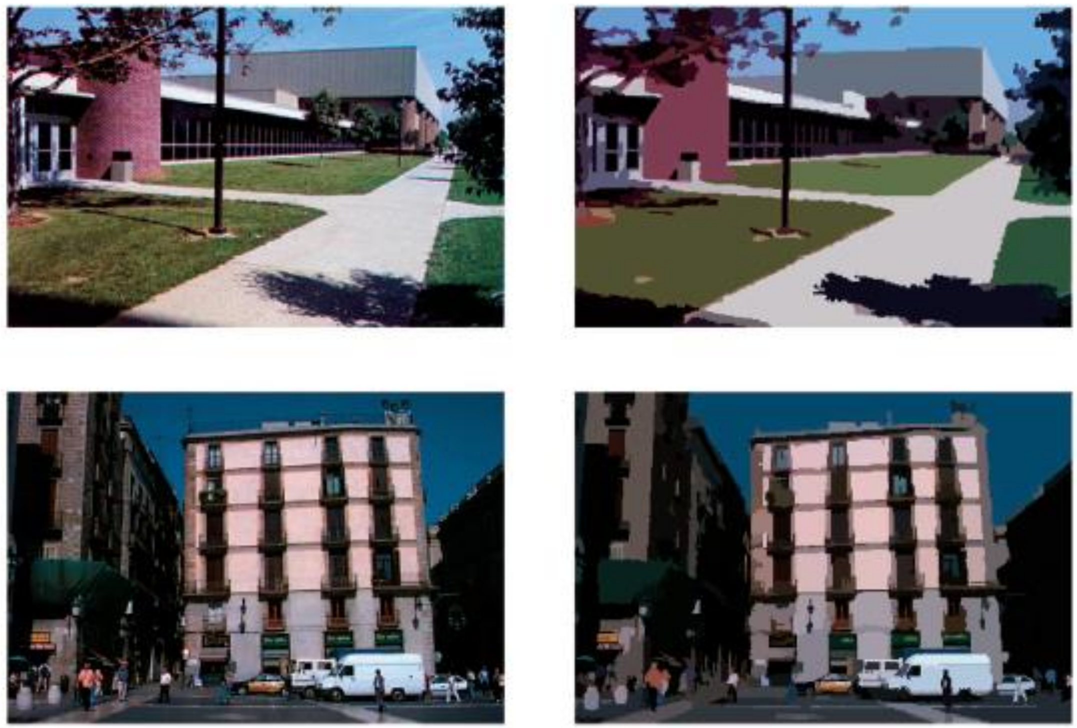
Segmentation

Example



Segmentation

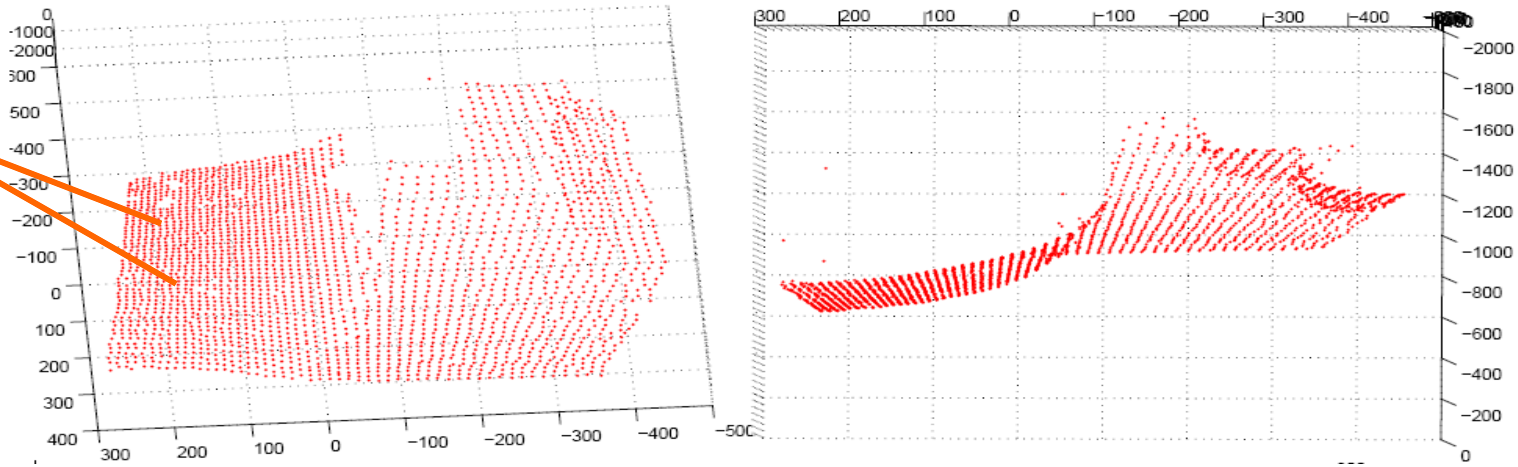
Example



Mean Shift applications:
N-D Segmentation

N-D segmentation

3D points



Feature space : Joint domain = 3D spatial coordinates + curvature + ...

$$K(\mathbf{x}) = C \cdot k_s\left(\left\|\frac{\mathbf{x}^s}{h_s}\right\|\right) \cdot k_r\left(\left\|\frac{\mathbf{x}^r}{h_r}\right\|\right)$$

Problem : How to choose the kernel bandwidth**s!**

Proposed Solution : A data driven stability criteria [Fukunaga 1990]

Stability criteria

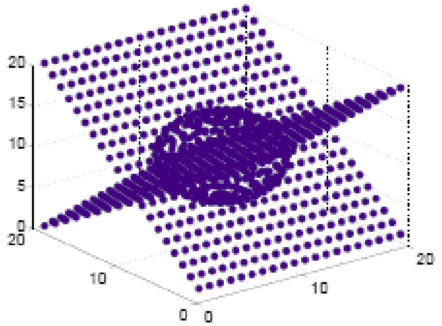
- 1. *Separate choice of the best bandwidth:*
  - for each sub-domain, perform MS clustering, using different increasing values of h.
  - After that, choose as best bandwidth value $h_{(\text{best})}$ the center of the largest operating range over which the same number of partitions are obtained for the given data.
- 2. *Final clustering*:
  - perform the mean shift clustering in the joint domain (position + curvature + etc.) using the kernel formed by concatenating the optimal sub-domain bandwidth values obtained in step 2)

$$\mathbf{h}_{(\mathbf{best})} = [h_{(\text{p,best})}\ h_{(\text{c,best})} \dots h_{(\text{etc, best})}]$$

Stability criteria - example

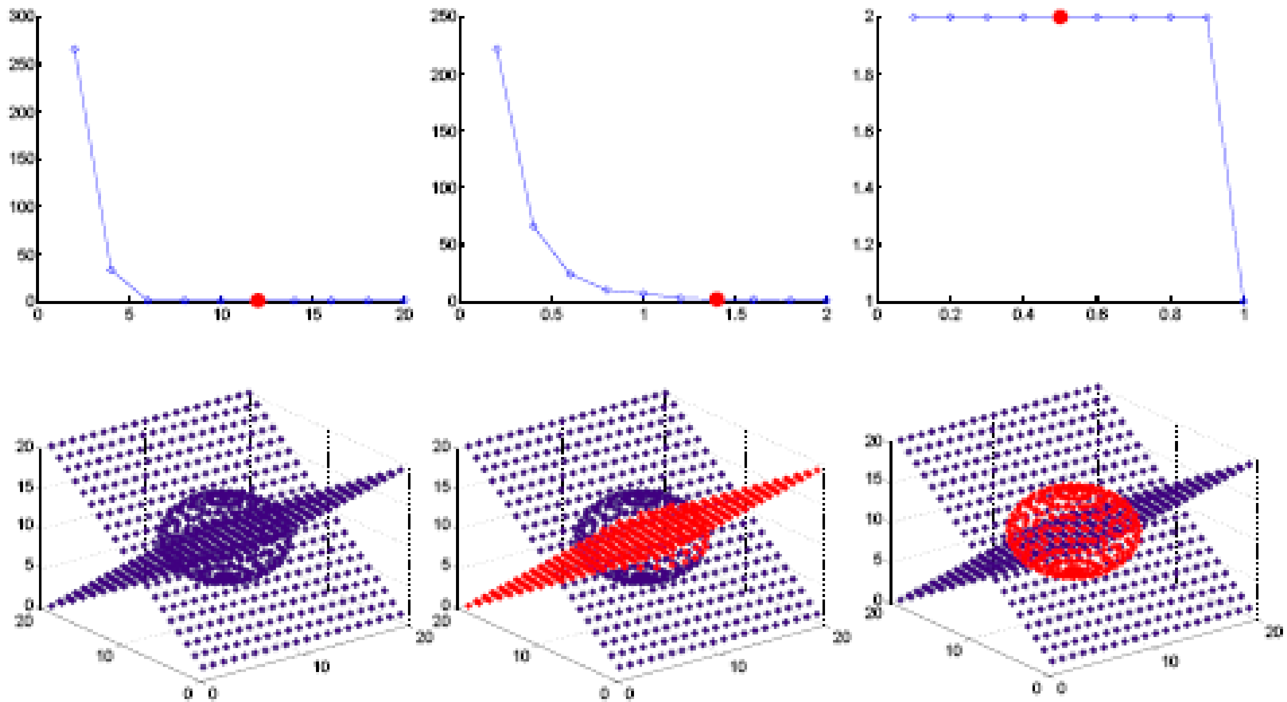
- Input: a set of data samples $x_i=[x_{i,s},x_{i,n},x_{i,c}]$
  - $x_{i,s}$: *spatial* coordinates
  - $x_{i,n}$: *normal* coordinates
  - $x_{i,c}$: *curvature* coordinates

- *Proposed algorithm*:

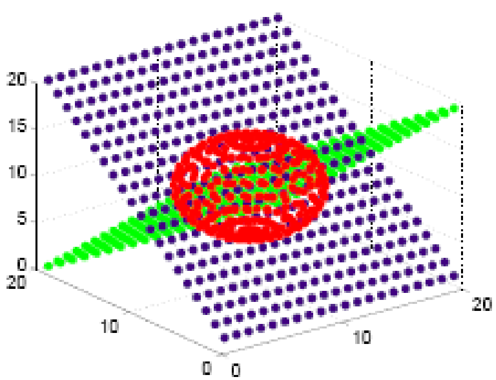
1) 

Standardization

Stability criteria - example



Separate choice of the best bandwidth



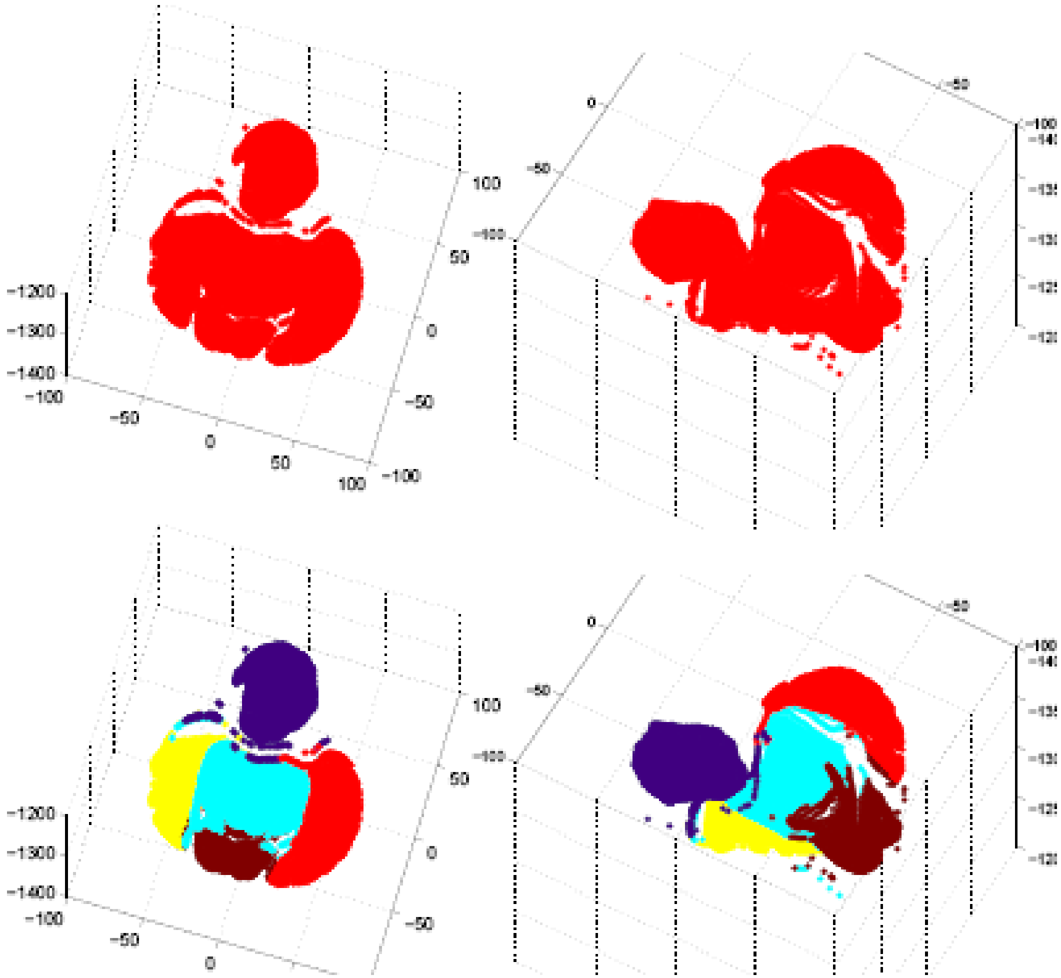
Final clustering

Stability criteria - real data results

Original

(Angel, Minolta dataset)

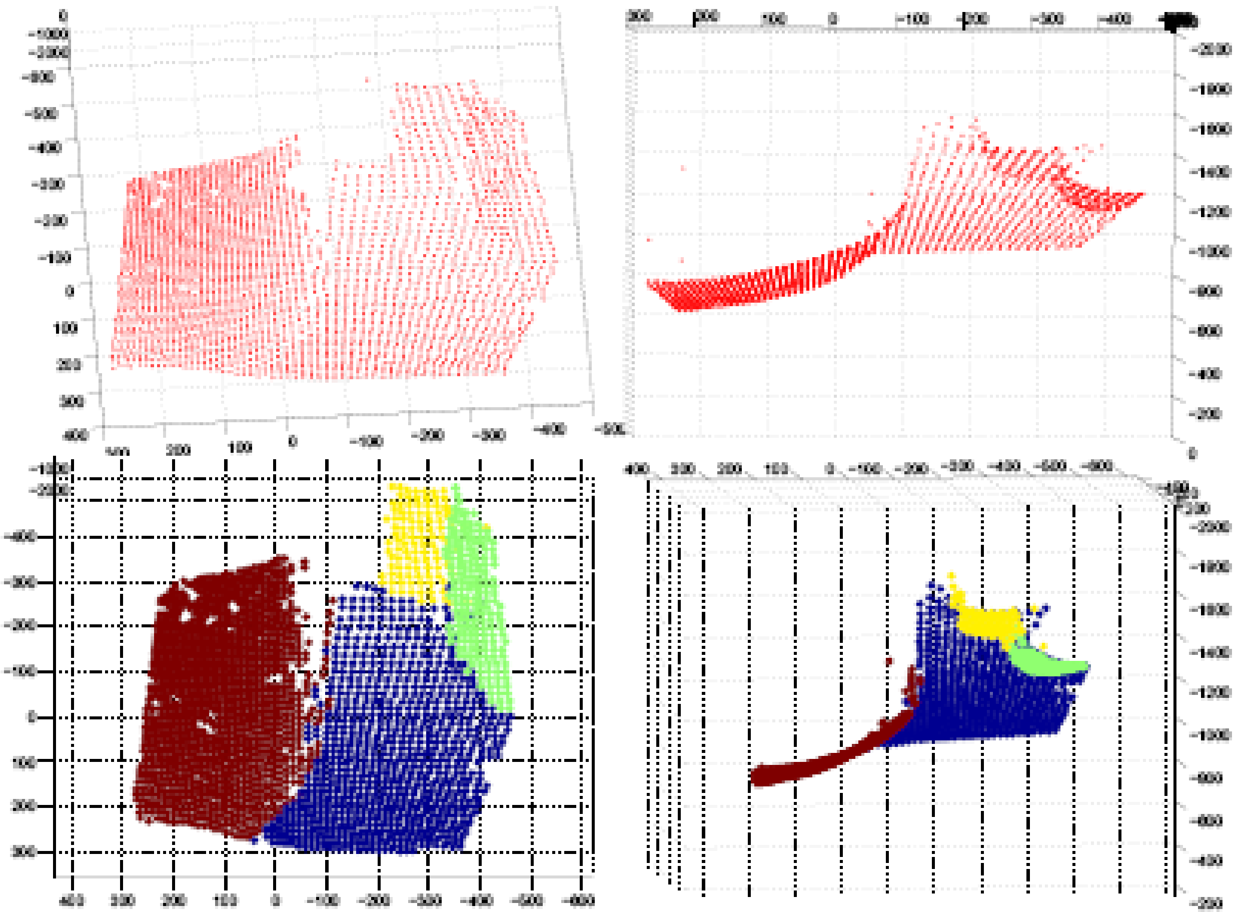
Result



Stability criteria - real data results

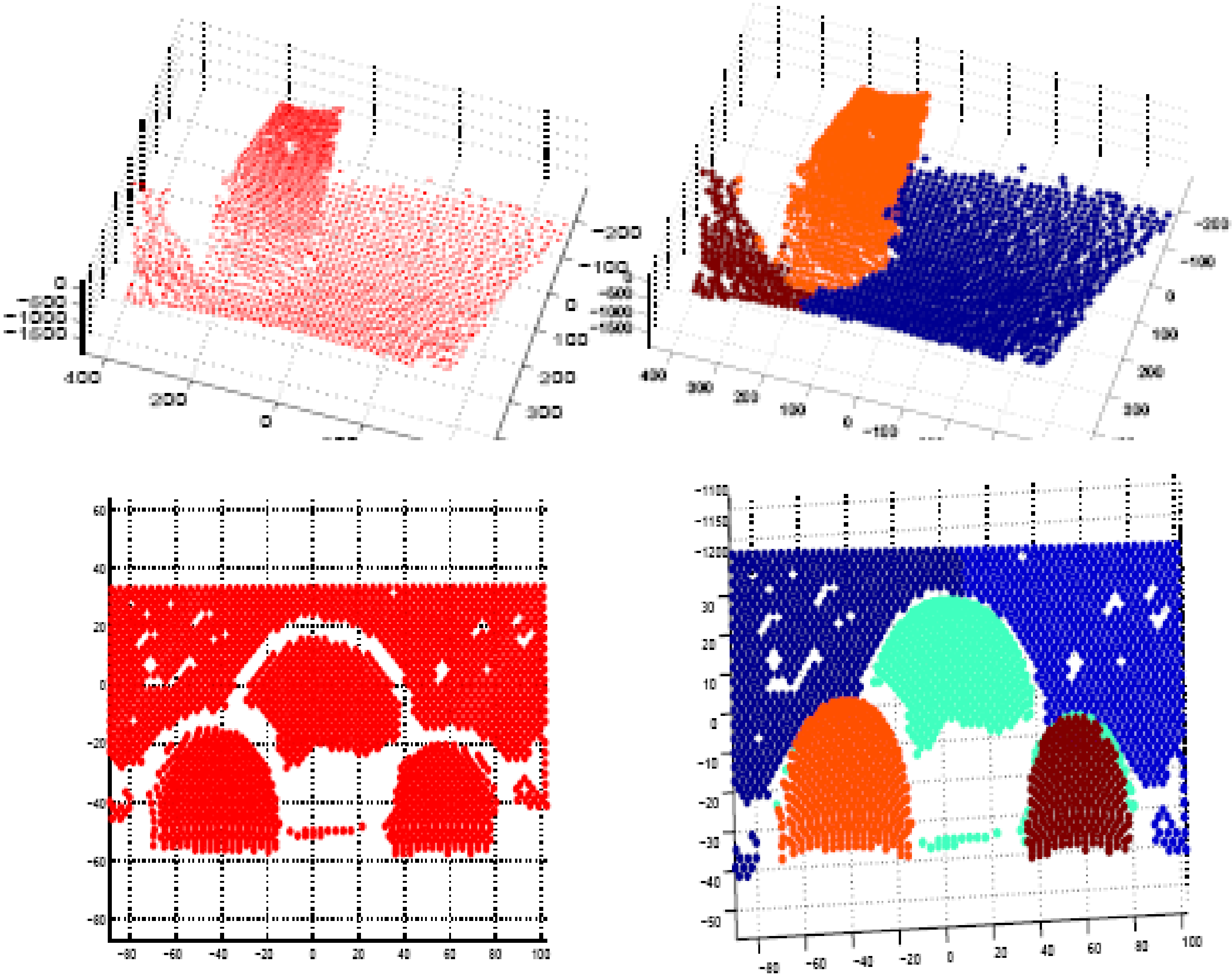
Original

(Acquired with echoscope sensor)



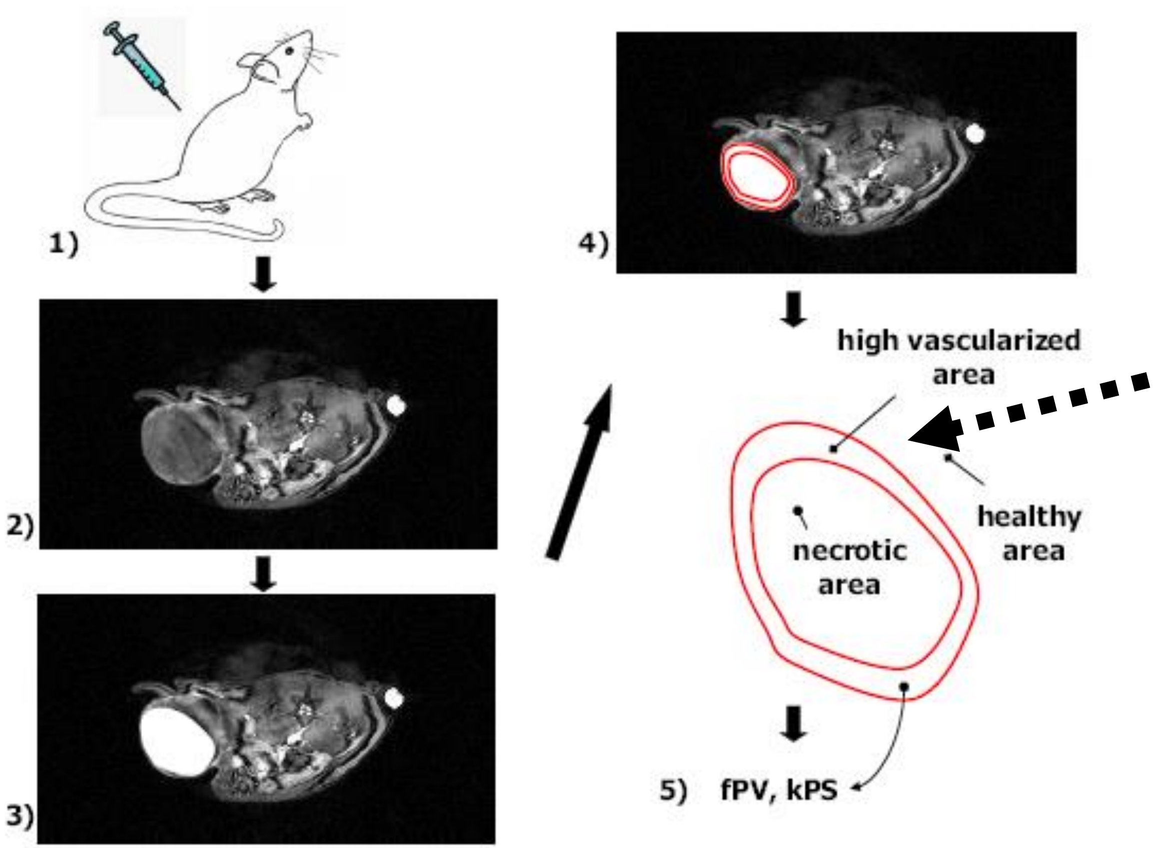
Result

Stability criteria - real data results



Another field of application: Medical-Imaging

The problem

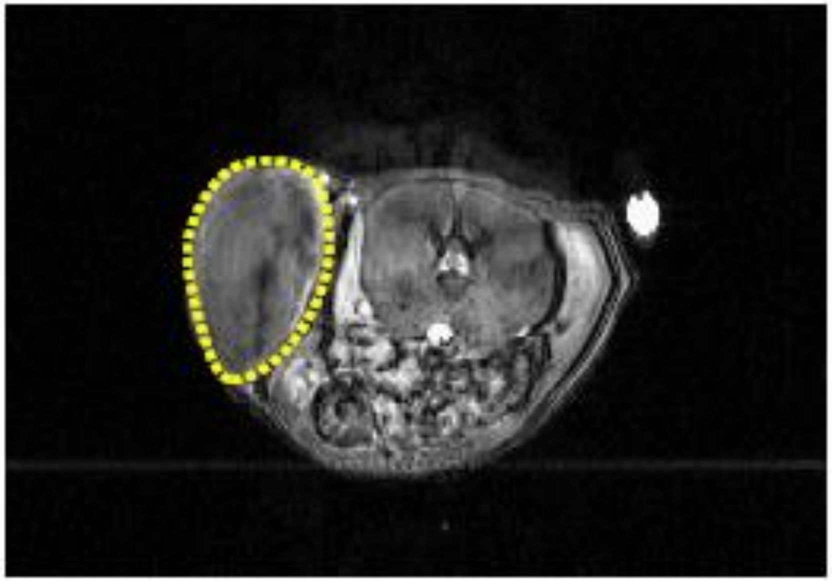


MANUALLY LABELED AREA!!

GOAL:

Automatize this process with Automatic Mean Shift

Another field of application: Medical-Imaging



Input

Result

Conclusions

- A robust modes estimation technique has been presented
- The technique is adaptive and non parametric
  - several applications
  - Only one *tuning parameter* to set is the kernel bandwidth
- We propose a data driven stability technique, that works well for N-D segmentations
- Application of our technique to other fields are currently under development (f.e. biomedical imaging

Publications

- M. Cristani, U. Castellani, V.Murino *Adaptive Feature Integration for Segmentation of 3D Data by Unsupervised Density Estimation*, Proceedings of Int'l Conf. on Pattern Recognition ICPR 2006, August 2006
- U. Castellani, M. Cristani, V.Murino *3D Data Segmentation Using a Non-Parametric Density Estimation Approach*, Proceedings of Eurographics Italian Chapter Conference '06 , pp.99-103, 2006.

END! (and thanks to Denis Simakov)